Multi-Source Contribution Learning for Domain Adaptation

Keqiuyin Li, Jie Lu, *Fellow, IEEE*, Hua Zuo, *Member, IEEE*, and Guangquan Zhang

***Abstract*— Transfer learning becomes an attractive technology to tackle a task from a target domain by leveraging previously acquired knowledge from a similar domain (source domain). Many existing transfer learning methods focus on learning one discriminator with single-source domain. Sometimes, knowledge from single-source domain might not be enough for predicting the target task. Thus, multiple source domains carrying richer transferable information are considered to complete the target task. Although there are some previous studies dealing with multi-source domain adaptation, these methods commonly combine source predictions by averaging source performances. Different source domains contain different transferable information; they may contribute differently to a target domain compared with each other. Hence, the source contribution should be taken into account when predicting a target task. In this article, we propose a novel multi-source contribution learning method for domain adaptation (MSCLDA). As proposed, the similarities and diversities of domains are learned simultaneously by extracting multi-view features. One view represents common features (similarities) among all domains. Other views represent different characteristics (diversities) in a target domain; each characteristic is expressed by features extracted in a source domain. Then multi-level distribution matching is employed to improve the transferability of latent features, aiming to reduce misclassification of boundary samples by maximizing discrepancy between different classes and minimizing discrepancy between the same classes. Concurrently, when completing a target task by combining source predictions, instead of averaging source predictions or weighting sources using normalized similarities, the original weights learned by normalizing similarities between source and target domains are adjusted using pseudo target labels to increase the disparities of weight values, which is desired to improve the performance of the final target predictor if the predictions of sources exist significant difference. Experiments on real-world visual data sets demonstrate the superiorities of our proposed method.**

***Index Terms*— Classification, deep learning, domain adaptation, transfer learning.**

I. Introduction

TRADITIONAL machine learning employs a great quantity of labeled data as a training set to learn a model, then applies the learned model to an unlabeled testing set which has the same feature space and probability distribution as the training data. However, in practice, it is rare to get sufficient labeled data of the same feature space or distribution for training because of its cost or issues concerning privacy, especially in industrial applications, which leads to the failure of traditional machine learning methods. Inspired by the learning mechanism where humans often tackle new tasks using methods gained from previous similar tasks, transfer learning [1] becomes an attractive choice for solving a current problem by transferring information from the labeled source domain to the unlabeled target domain.

Based on different settings, transfer learning [2] is divided into three cases: inductive transfer learning, including multi-task and self-taught learning, where target domain labels are available; transductive transfer learning, including domain adaptation, where target domain labeling is unavailable but source domain labels are; and unsupervised transfer learning, including clustering, where both source and target domain labels are unavailable. Domain adaptation [3] is a popular method used to achieve cross-domain transformation which completes the prediction task in a target domain with the assistance of previously gained knowledge from a similar source domain. According to the number of source and target domains, it falls within the cases of single-source and single target [4], multi-source and single target [5], [6], single-source and multi-target [7], and multi-source and multi-target domain adaptation.

To achieve domain adaptation, since original source and target features can be homogeneous or heterogeneous [8]–[12] and follow different distributions, one solution is to transform them into a latent feature space. Extracting latent features via reducing discrepancy of source and target domains is a common procedure. Taking advantage of deep learning, convolutional neural networks (CNNs), such as AlexNet [13] and ResNet [14] pretrained on ImageNet, are widely used to transform source and target data into a latent feature space and extract robust representations [15]–[17] for visual domain adaptation.

Single-source and single target domain adaptation has been widely explored in previous research [18]–[21]. Since scarcely any of previous studies focus on the multi-target domain adaptation, here we directly discuss the multi-source and single target domain adaptation. Recently, multi-source domain adaptation has attracted increasing attention since it can provide richer information compared with single-source domain adaptation. On the other hand, it also introduces a major challenge to this research field because of domain shifts, which means we cannot simply combine all source domains as

Manuscript received July 16, 2020; revised December 31, 2020; accepted March 25, 2021. This work was supported by the Australian Research Council under Grant FL190100149. *(Corresponding author: Jie Lu.)*

The authors are with the Centre for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: keqiuyin.li@student.uts.edu.au; jie.lu@uts.edu.au; hua.zuo@uts.edu.au; guangquan.zhang@uts.edu.au).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3069982.

Digital Object Identifier 10.1109/TNNLS.2021.3069982

one [22], [23]. Some methods are developed to tackle multi-source domain adaptation [24]–[27], while most previous multi-source domain adaptation methods complete a target predictor by averaging all source predictions without due consideration of their different contributions. Although weighted combination is employed [28]–[30], in fact, the weights with minor different quantities might lead to parallel performance as averaged combination, and it might be invalid where the contributions of sources have significant difference, which can result in negative transfer. Negative transfer is a fairly common phenomenon, but identifying when and where it occurs is both difficult and challenging, and there is still no effective way of identifying it. To measure contributions of multiple sources and reduce the degrading influence of negative transfer which harms the final performance of target predictor, we propose a weight learning method with pseudo labels for multi-source domain adaptation. The proposed framework adapts all source and target domains simultaneously by minimizing their discrepancies. At the same time, since the target domain might contain diverse characteristics which can be represented by different source domains, the diversities of domains are learned by maximizing their discrepancies. To measure the discrepancy between two domains, both domain-level and class-level discrepancies are considered. Our contributions can be summarized as threefold.

1) Development of a new method to learn weights of source domains using their predicted pseudo labels of target domain. The learned weights are then applied to complete the target predictor, which can take advantage of the best performing source domain. In this way, it will guarantee the target performance if source predictions exhibit significant difference.
2) A representation extraction framework to explore the similarities and the diversities among all source and target domains, which enriches transfer information by providing multiple views of common and specific features. This is valuable when we come to explore target features from multiples aspects and extract comprehensive information, many existing studies only focus on single-view features.
3) An alignment structure to learn the similarities between source and target domains by measuring domain-level and class-level discrepancies simultaneously, which undermines the misalignment of boundary samples. It can enlarge the category distance and reduce the influence of cluster boundaries.

The remainder of this article is designed as follows. Section II briefly describes some technologies for domain adaptation used in previous studies. Section III details the method we propose. Experiments carried on real-world visual data sets are presented in Section IV. Section V concludes the whole work and formulates potential future work.

II. Related Work

This section introduces definitions and previous studies on domain adaptation, listing in the main common techniques of measuring discrepancy between two domains, single-source domain adaptation, different levels of distribution matching, multi-source domain adaptation methods, and relevant combination rules.

A. Discrepancy Measuring Technology

Domain adaptation is a special category of transfer learning which learns from source data and transfers common knowledge across different but related target data. That is learning a target task using an obtained source model by overcoming domain shifts. The main step is exploring the similarities between source and target domains via reducing discrepancy. Many technologies have been proposed to measure discrepancy between two domains. Maximum mean discrepancy (MMD) [31] is a widely used method to test if two samples are drawn from the same distribution.

Transfer component analysis [4] employs MMD to achieve marginal distribution adaptation. Deep domain confusion [32] incorporates MMD into deep networks. Based on these, deep adaptation network [15] extends multi-kernel MMD [33] to adapt deep representations. Joint adaptation network [34] proposes joint MMD to align joint distributions of multiple domain layers. Contrastive domain discrepancy [35] and deep-kernel based MMD [36] have been extended recently to enhance the transferability of latent features. Central moment discrepancy [37] is proposed to reduce the computational complexity of minimizing domain-specific latent representations compared with MMD. Some domain adaptation methods employ Wasserstein distance [30], [38], [39] and adversarial learning [40], [41] to obtain more stable and robust gradients in the situation where the distributions of source and target domains have no overlaps. The graph-matching metric [42], [43] is developed as the domain discrepancy measurement that has the ability to map both nodes and edges between source and target representations. By doing this, not only is distribution knowledge considered, but also structural and geometric information which is rarely investigated in most previous studies is considered.

B. Single-Source Domain Adaptation

Multi-representation adaptation network [18] aligns the distributions of source and target representations by a multi-structure network which extracts representations from different aspects. Joint geometrical and statistical alignment [19] is presented to unify shared space and subspaces of source and target domains via reducing the shifts of the geometries as well as the distributions simultaneously. Asymmetric tri-training [20] trains multiple classifiers using labeled data from source domain and then generates and updates artificial labels of unlabeled samples to learn target networks.

All above-mentioned measures are built on the assumption that matching different distributions in the transformed latent feature spaces can lead to success in domain adaptation. Saito *et al.* [20], however, believe sometimes a well-performed predictor on both source and target domains may not exist even when their distributions are matched [44], and develop a method to adapt source and target domains by assigning pseudo-labeled target samples and labeled source samples instead of distributions. CycleGAN [45], [46] is now

widely employed to generate target samples and use them to train a target network directly with a pretrained network on source domain. To deal with domain adaptation where the source data are unavailable, fine-tuning-based methods [47], [48] are developed to match model parameters of source and target networks. Source hypothesis transfer [48] learns the target predictor by freezing classifier module but fine-tuning feature extraction module without the access of original source data.

C. Multi-Level Distribution Matching

Except for reducing the discrepancy of source and target domains on domain-level alone, various kinds of multiple-level distributions matching are explored to improve transfer performance. Dynamic adversarial adaptation network [41] assesses different contributions of marginal and conditional distributions between domains dynamically. Transferable attention network [49] diminishes multiple region-level and single image-level distribution discrepancies. Multi-adversarial domain adaptation matches domain-level and class-level distributions by multi-mode discriminators. Local feature pattern method [50] jointly maps holistic feature distribution and local pattern distributions. These multi-level distribution matching technologies enable fine-grained alignment of cross-domain adaptation.

D. Multi-Source Domain Adaptation and Combination Rules

To achieve the desired performance of the target predictor, multi-domain matching network [24] matches distributions of different domains by extracting relationships within sources as well as among sources and target simultaneously. Multi-source domain adversarial network [25] provides generalization bounds which can be used to extract domain invariant and task discriminative features simultaneously under both classification and regression settings. Moment matching for multi-source domain adaptation [29], a method tested on a new created multi-domain data set, aligns the sources with each other and with the target during training using adversarial learning. Domain generalization with adversarial feature learning [51] extracts domain-invariant representations from multiple source domains to learn a universal classifier to be performed on an unseen target domain, where there is entirely no target data available during training. Mixture of multiple latent domains for domain generalization [52] is developed to tackle a novel case where the label of a sample belonging to a domain is unknown, and adversarial learning is employed to extract shared features from pseudo-labeled domains divided by clustering.

For multi-source domain adaptation, combining source predictors to complete the target task is an essential step. A common and simple method here is averaging all source performances. Multiple feature spaces adaptation network [26] aims to align domain-specific distributions as well as domain-specific classifiers to reduce misclassification caused by samples near class boundaries. However, different sources commonly deliver different contributions, which means the weights of sources may need to be designed rather than averaging.

To determine the combined weights of source domains, the multi-source selective transfer method [53] builds three selection strategies including nearest selection, weighted selection, and Top-k selection to choose closer source domains to target domain. The deep cocktail network [28] focuses on multi-source domain adaptation where each source domain shares partial categories with the target domain using a multi-path adversarial learning method. And it employs multiple source-target-specific perplexity scores to re-weight contributions of source predictions. Multi-source distilling domain adaptation [30] selects training source samples which are closer to target samples and distills dissimilar samples to define a novel discrepancy-based strategy to learn weights under a presumption that the estimated similarity between each source and target domains follows standard Gaussian distribution. Multi-source adversarial domain aggregation network [54], instead of learning weights of source domains, trains one model by adversarial domain aggregation which closely combines all adapted domains together.

Our work is based mainly on the network structure proposed in multiple feature spaces adaptation network [26], the differences being that we measure domain-level similarity/diversity between each source and target domains as well as within source domains using multi-view features, at the same time, class-level similarity is also considered. The learned similarity is then used to evaluate the contributions of sources and weight source predictions with their estimated pseudo labels. The whole framework of our proposed method is outlined in Fig. 1. Comparison with some other state-of-the-art domain adaptation methods indicates the superiority of the method we propose.

III. Proposed Method

The proposed method contains three parts: multi-view feature extraction, multi-level distribution matching, and source-specific predictors and target predictor learning. As showed in Fig. 1, feature extraction entails extracting features that are shared, common, and diverse. Shared features are obtained using pretrained networks before being divided into common and diverse features. The former represents common knowledge across all domains, and the latter specifies knowledge shared by each source and target domains. This approach is expected to express target domain from different perspectives and provide richer information for completing the target task. These extracted multi-view features are then fed into distribution matching, where domain-level matching is employed to adapt source and target features, while class-level matching reduces the misalignment of boundary samples. Source predictors are learned using matched features of source and target domains, while the target predictor is completed by combining source predictors with adjusted weights, which is chosen to reduce negative transfer.

A. Problem Setting and Notations

We focus on homogeneous unsupervised multi-source domain adaptation, where the feature spaces of labeled source

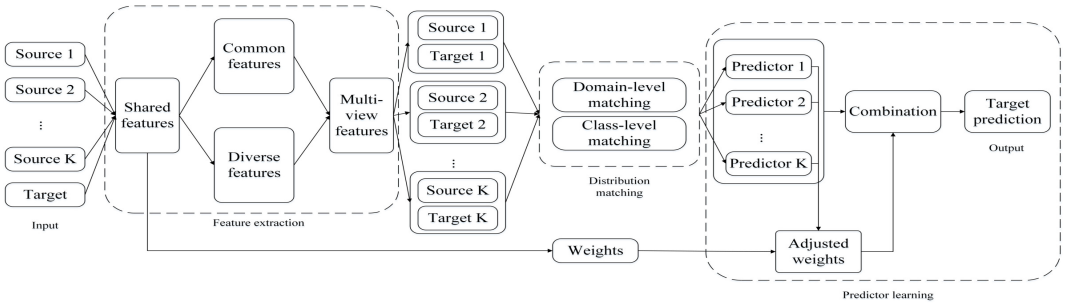
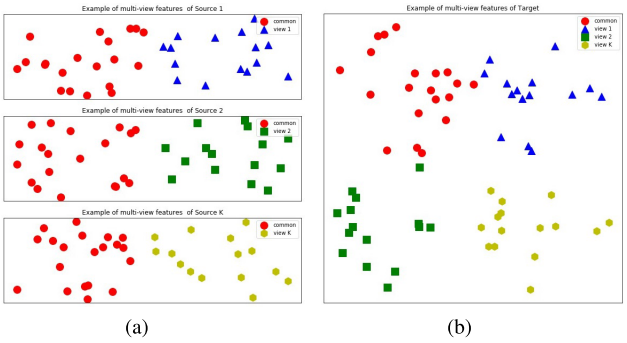


Fig. 1. Whole framework of the proposed method. Shared features are collected using pretrained networks. Common features represent similarities among all source and target domains, while diverse features represent diversities contained in the target domain which can be expressed by different source domains. Target k means features of target collected using kth source features extraction networks.

domains and unlabeled target domain have the same dimension. Given K labeled source domains $\{\mathcal{D}_{s_k}\}_{k=1}^{K}$, for each domain $\mathcal{D}_{s_k} = \{(\boldsymbol{X}_{s_k}, Y_{s_k})\} = \{(\boldsymbol{x}_{s_k}^i, \boldsymbol{y}_{s_k}^i)\}_{i=1}^{n_{s_k}}$, where $\boldsymbol{X}_{s_k} \in \mathcal{X}$ represents observed samples which follow distribution $\mathcal{P}_{s_k}$ and $\boldsymbol{Y}_{s_k} \in \mathcal{Y}$ indicates corresponding labels of $\boldsymbol{X}_{s_k}$, $\mathcal{X}, \mathcal{Y}$ indicate original data space and label set, and n_s indicates the number of samples in each source domain. The unlabeled target domain is represented as $\mathcal{D}_t = \{\boldsymbol{X}_t\} = \{\boldsymbol{x}_t^j\}_{j=1}^{n_t}$, where $\boldsymbol{X}_t \in \mathcal{X}$ follows distribution $\mathcal{P}_t$ and n_t is the number of samples. All source and target domains share the same categories, which means the predicted target label $\boldsymbol{Y}_t \in \mathcal{Y}$. We apply the proposed method mainly to image classification.



(a) (b)

Fig. 2. Example of characteristics contained in source and target domains. (a) Source domain. (b) Target domain.

B. Multi-View Feature Extracting

Aiming to extract latent features of source and target domains for adapting, a pretrained deep neural network ϕ is first used to transform the original data into a shared feature space. The transformation can be expressed as

$$\begin{aligned} \boldsymbol{f}_{s_k}^i &= \phi(\boldsymbol{x}_{s_k}^i, \boldsymbol{\theta}), \\ \boldsymbol{f}_t^j &= \phi(\boldsymbol{x}_t^j, \boldsymbol{\theta}), \\ i = 1, 2, \ldots, n_{s_k}, j &= 1, 2, \ldots, n_t, k = 1, 2, \ldots, K, \end{aligned} \tag{1}$$

where $\boldsymbol{f}_{s_k}, \boldsymbol{f}_t$ represent features in shared feature space, $\boldsymbol{\theta}$ means parameters of deep network ϕ.

Normally, a picture shows features from multiple views such as text, edge, chrominance, luminance, and so on. Based on this fact, the target domain may have multiple aspect characteristics that each view of these characteristics can be reflected by a source domain more similarly compared with other source domains. For example, in data set Office-Home [55], domain clipart might resemble domain art more on image text since they all have artistic pictures. At the same time, it might be more similar to domain product on image edge because they are all without background. Fig. 2 shows an example of diverse characteristics contained in source and target domains generated using synthetic data. Assuming each shape in the figure indicates a different characteristic of the target domain, and if this characteristic can be extracted as one kind of feature in latent space, what we expect is to find that characteristic in the source domain which is similar to that from target domain. One source domain might contain partial views, and the union of all source domains might cover all view features of target domain.

Taking the described factor into consideration, the collected shared features are then split into two parts to represent the target domain more completely. One part carries common transferable information, and the other holds diverse transferable information. The common feature extraction can be represented as

$$\begin{aligned} \boldsymbol{f}_{c_{s_k}}^i &= \phi_{c_k}(\boldsymbol{f}_{s_k}^i, \boldsymbol{\theta}_{c_k}), \\ \boldsymbol{f}_{c_{t_k}}^j &= \phi_{c_k}(\boldsymbol{f}_t^j, \boldsymbol{\theta}_{c_k}), \\ i = 1, 2, \ldots, n_{s_k}, j &= 1, 2, \ldots, n_t, k = 1, 2, \ldots, K, \end{aligned} \tag{2}$$

while the diverse feature extraction is

$$\begin{aligned} \boldsymbol{f}_{d_{s_k}}^i &= \phi_{d_k}(\boldsymbol{f}_{s_k}^i, \boldsymbol{\theta}_{d_k}), \\ \boldsymbol{f}_{d_{t_k}}^j &= \phi_{d_k}(\boldsymbol{f}_t^j, \boldsymbol{\theta}_{d_k}), \\ i = 1, 2, \ldots, n_{s_k}, j &= 1, 2, \ldots, n_t, k = 1, 2, \ldots, K, \end{aligned} \tag{3}$$

where ϕ_{c_k}, ϕ_{d_k} mean feature extractors of kth source domain, and $\boldsymbol{\theta}_{c_k}, \boldsymbol{\theta}_{d_k}$ are corresponding parameters. Each view diverse features can be homogeneous or heterogeneous compared with other views, which means the structures of $\{\phi_{d_k}\}_{k=1}^{K}$ can be different. Besides, ϕ_{c_k} and ϕ_{d_k} can distill redundancy information to some degree by reducing the dimension of

shared features $\boldsymbol{f}_{s_k}$ and $\boldsymbol{f}_t$. This dimension reduction is widely used in domain adaptation.

C. Multi-Level Distribution Adapting

The common features extracting processing is controlled by minimizing any discrepancy of common features among all domains, including within source domains and between each source and target domains. Since the target domain has multi-view features, we first adapt source domains from the common view, while the adaptation of sources and target will be done later with the diverse features. Here we choose MMD as the discrepancy measure, measuring the loss function of source common features extraction. It can be written as

$$\begin{aligned}\mathcal{L}_c &= \frac{2}{K(K-1)}\sum_{k_1=1}^{K-1}\sum_{k_2=k_1+1}^{K}\mathcal{MMD}(\boldsymbol{X}_{s_{k_1}},\boldsymbol{X}_{s_{k_2}})\\ &= \frac{2}{K(K-1)}\sum_{k_1=1}^{K-1}\sum_{k_2=k_1+1}^{K}\\ &\quad\left\|\frac{1}{n_{s_{k_1}}}\sum_{i=1}^{n_{s_{k_1}}}\mathcal{K}\left(\boldsymbol{f}^i_{c_{s_{k_1}}}\right)-\frac{1}{n_{s_{k_2}}}\sum_{j=1}^{n_{s_{k_2}}}\mathcal{K}\left(\boldsymbol{f}^j_{c_{s_{k_2}}}\right)\right\|^2_{\mathcal{H}}\end{aligned}\tag{4}$$

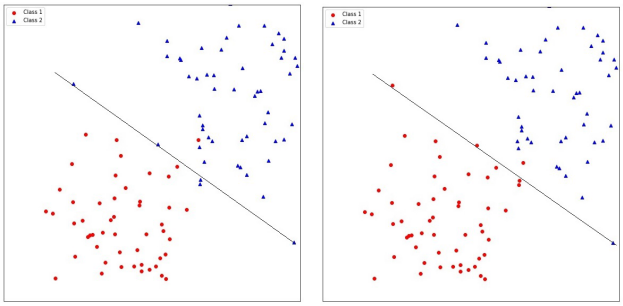
where $\|\cdot\|_{\mathcal{H}}$ indicates the reproducing kernel Hillbert space (RKHS) norm, and $\mathcal{K}$ is kernel-induced feature transformation. During training, the number of samples, $n_{s_{k_2}}$ and $n_{s_{k_2}}$, can be replaced with batch size. This operation is applicable to all MMD calculations in this article.

For diverse views, a preferred solution is training a multiple structure networks to extract these features from the target domain directly. However, since the target data are unlabeled, this entirely unsupervised collecting of features for a target task without any assistance rarely meets requirement. Considering this, it can be adapted for extracting diverse features of source domains by maximizing the discrepancy of sources and matching distributions of sources and target simultaneously from diverse views. This can avoid the high correlation between common and diverse features at the same time. The source diverse features extraction loss function is

$$\begin{aligned}\mathcal{L}_d &= \frac{2}{K(K-1)}\sum_{k_1=1}^{K-1}\sum_{k_2=k_1+1}^{K}\mathcal{MMD}(\boldsymbol{X}_{s_{k_1}},\boldsymbol{X}_{s_{k_2}})\\ &= \frac{2}{K(K-1)}\sum_{k_1=1}^{K-1}\sum_{k_2=k_1+1}^{K}\\ &\quad\left\|\frac{1}{n_{s_{k_1}}}\sum_{i=1}^{n_{s_{k_1}}}\mathcal{K}\left(\boldsymbol{F}^i_{d_{s_{k_1}}}\right)-\frac{1}{n_{s_{k_2}}}\sum_{j=1}^{n_{s_{k_2}}}\mathcal{K}\left(\boldsymbol{F}^j_{d_{s_{k_2}}}\right)\right\|^2_{\mathcal{H}}\end{aligned}\tag{5}$$

where $\{\boldsymbol{F}_{d_{s_{k_1}}},\boldsymbol{F}_{d_{s_{k_2}}}\}=\{\boldsymbol{f}_{d_{s_{k_1}}},\boldsymbol{f}_{d_{s_{k_2}}}\}$, if $\boldsymbol{f}_{d_{s_{k_1}}},\boldsymbol{f}_{d_{s_{k_2}}}\in\mathbb{R}^m$, which means homogeneous. $\boldsymbol{F}_{d_{s_{k_1}}}=[\boldsymbol{f}_{d_{s_{k_1}}};\boldsymbol{O}^{m_2}]$ and $\boldsymbol{F}_{d_{s_{k_2}}}=[\boldsymbol{O}^{m_1};\boldsymbol{f}_{d_{s_{k_2}}}]$ if $\boldsymbol{f}_{d_{s_{k_1}}}\in\mathbb{R}^{m_1}$ and $\boldsymbol{f}_{d_{s_{k_2}}}\in\mathbb{R}^{m_2}, m_1\neq m_2$, which means heterogeneous, $\boldsymbol{O}$ is a null matrix. As mentioned before, in this work, we only explore the homogeneous setting but might focus on a heterogeneous setting as future work. Then the total loss of source domains adaptation processing is

$$\mathcal{L}_s=\mathcal{L}_c-\mathcal{L}_d.\tag{6}$$



(a) (b)

Fig. 3. Example of boundary samples. (a) Original classes. (b) Learning classes.

The extracting of these features for each source domain is controlled by mapping source and target distributions. For each source domain $\mathcal{D}_{s_k}$, the domain-level distribution matching is

$$\begin{aligned}\mathcal{L}_{\text{domain}} &= \mathcal{MMD}(\boldsymbol{X}_{s_k},\boldsymbol{X}_t)\\ &= \left\|\frac{1}{n_{s_k}}\sum_{i=1}^{n_{s_k}}\mathcal{K}\left(\boldsymbol{F}^i_{\text{cat}_{s_k}}\right)-\frac{1}{n_t}\sum_{j=1}^{n_t}\mathcal{K}\left(\boldsymbol{F}^j_{\text{cat}_{t_k}}\right)\right\|^2_{\mathcal{H}}\end{aligned}\tag{7}$$

where $\boldsymbol{F}_{cat_{s_k}}=[\boldsymbol{f}_{c_{s_k}};\boldsymbol{f}_{d_{s_k}}]$, $\boldsymbol{F}_{cat_{d_k}}=[\boldsymbol{f}_{c_{t_k}};\boldsymbol{f}_{d_{t_k}}]$.

Except for adapting each source and target on domain level, to reduce the misalignment of boundary samples, we also consider the class-level distribution matching. A simple synthetic example of boundary samples is given in Fig. 3. If the black line is the classifier, samples around it may attract wrong labels. For most complex classification tasks, softmax function is a widely used technology to compute the probabilities of a sample belonging to all classes and to choose the maximal one as its final label. However, samples near class boundaries may get the same probabilities of different classes or a wrong maximal class probability, so we consider maximizing discrepancy among different classes and minimizing the discrepancy within the same classes to solve this problem.

The class-level distribution matching is controlled by

$$\begin{aligned}\mathcal{L}_{class} &= \frac{1}{C}\sum_{r=1}^{C}\mathcal{MMD}\left(\boldsymbol{X}^r_{s_k},\boldsymbol{X}^r_t\right)\\ &\quad-\left(\frac{2\lambda}{3C(C-1)}\sum_{r_1=1}^{C-1}\sum_{r_2=r_1+1}^{C}\right.\\ &\qquad\left.\left(\mathcal{MMD}\left(\boldsymbol{X}^{r_1}_{s_k},\boldsymbol{X}^{r_2}_{s_k}\right)+\mathcal{MMD}\left(\boldsymbol{X}^{r_1}_t,\boldsymbol{X}^{r_2}_t\right)\right)\right)\\ &\quad-\frac{\lambda}{3C(C-1)}\sum_{r_s=1}^{C}\sum_{r_t\neq r_s}^{C}\mathcal{MMD}\left(\boldsymbol{X}^{r_s}_{s_k},\boldsymbol{X}^{r_t}_t\right).\end{aligned}\tag{8}$$

Symbols r, n with superscripts or subscripts in above equations indicate corresponding class index, number of features in each class, respectively, C is the total categories of domains, $\lambda\in[0,1]$ is a trade-off constant evaluating the contribution of interclass discrepancy.

From (8), if all interclass discrepancies are calculated, the computation will be high especially when the class number C is large. In practice, the boundary samples are frequently misclassified between two nearest categories. To reduce computation extent, (8) can be rewritten as below, where we only

maximizes the margin of the nearest two classes within each source and target domains:

$$\mathcal{L}_{\text{class}} = \frac{1}{C}\sum_{r=1}^{C}\mathcal{MMD}\left(\boldsymbol{X}_{s_k}^{r}, \boldsymbol{X}_{t}^{r}\right) - \left(\frac{\lambda}{2}\left(\mathcal{MMD}\left(\boldsymbol{X}_{s_k}^{r_{s1}}, \boldsymbol{X}_{s_k}^{r_{s2}}\right) + \mathcal{MMD}\left(\boldsymbol{X}_{t}^{r_{t1}}, \boldsymbol{X}_{t}^{r_{t2}}\right)\right)\right) \quad (9)$$

where:

$$\mathcal{MMD}\left(\boldsymbol{X}_{s_k}^{r}, \boldsymbol{X}_{t}^{r}\right) = \left\| \frac{1}{n_{s_k}^{r}}\sum_{i=1}^{n_{s_k}^{r}}\mathcal{K}\left(p_{s_k}^{ir}\cdot\boldsymbol{F}_{\text{cat}_{s_k}}^{ir}\right) - \frac{1}{n_{t_k}^{r}}\sum_{j=1}^{n_{t_k}^{r}}\mathcal{K}\left(p_{t_k}^{jr}\cdot\boldsymbol{F}_{\text{cat}_{t_k}}^{jr}\right)\right\|_{\mathcal{H}}^{2}$$

$$\mathcal{MMD}\left(\boldsymbol{X}_{s_k}^{r_{s1}}, \boldsymbol{X}_{s_k}^{r_{s2}}\right)) = \left\| \frac{1}{n_{s_k}^{r_{s1}}}\sum_{i=1}^{n_{s_k}^{r_{s1}}}\mathcal{K}\left(p_{s_k}^{ir_{s1}}\cdot\boldsymbol{F}_{\text{cat}_{s_k}}^{ir_{s1}}\right) - \frac{1}{n_{s_k}^{r_{s2}}}\sum_{j=1}^{n_{s_k}^{r_{s2}}}\mathcal{K}\left(p_{s_k}^{jr_{s2}}\cdot\boldsymbol{F}_{\text{cat}_{s_k}}^{jr_{s2}}\right)\right\|_{\mathcal{H}}^{2}$$

$$\mathcal{MMD}\left(\boldsymbol{X}_{t}^{r_{t1}}, \boldsymbol{X}_{t}^{r_{t2}}\right) = \left\| \frac{1}{n_{t_k}^{r_{t1}}}\sum_{i=1}^{n_{t_k}^{r_{t1}}}\mathcal{K}\left(p_{t_k}^{ir_{t1}}\cdot\boldsymbol{F}_{\text{cat}_{t_k}}^{ir_{t1}}\right) - \frac{1}{n_{t_k}^{r_{t2}}}\sum_{j=1}^{n_{t_k}^{r_{t2}}}\mathcal{K}\left(p_{t_k}^{jr_{t2}}\cdot\boldsymbol{F}_{\text{cat}_{t_k}}^{jr_{t2}}\right)\right\|_{\mathcal{H}}^{2}.$$

p is the probability of a sample belonging to class r, subscript $s1, t1$ indicate the maximal class probabilities, and $s2, t2$ represent the second maximal class probabilities. The total loss of target domain adaptation is

$$\mathcal{L}_t = \mathcal{L}_{\text{domain}} + \mathcal{L}_{\text{class}}. \quad (10)$$

D. Predictions Learning

After adapting all source and target domains, the predictors of source tasks can be learned and applied to the target task. Cross entropy is employed to optimize the predictors, for each source domain $\mathcal{D}_k$, it can be represented as

$$\mathcal{L}_p = -\frac{1}{n_{s_k}}\sum_{i=1}^{n_{s_k}} y_{s_k}^{i}\log\left(P_{s_k}\left(\boldsymbol{F}_{\text{cat}_{s_k}}^{i}\right)\right). \quad (11)$$

P_{s_k} is the predictor of kth source domain. When applying the learned source predictors to the target task, it is desired that all source predictors could return the same results of the same target samples. So the cross-domain constraint is added to minimize errors of different predictions on the same target samples

$$\mathcal{L}_{\text{cro}} = \frac{2}{K(K-1)}\sum_{k_1=1}^{K-1}\sum_{k_2=k_1+1}^{K}\left(\frac{1}{n_t}\sum_{j=1}^{n_t}\left|P_{s_{k_1}}\left(\boldsymbol{F}_{\text{cat}_{t_{k_1}}}^{j}\right) - P_{s_{k_2}}\left(\boldsymbol{F}_{\text{cat}_{t_{k_2}}}^{j}\right)\right|\right). \quad (12)$$

The total loss of predictor learning of each source is

$$\mathcal{L} = \mathcal{L}_p + \alpha\mathcal{L}_s + \beta\mathcal{L}_t + \gamma\,\mathcal{L}_{\text{cro}} \quad (13)$$

α, β, γ are trade-off parameters.

To complete target task with multiple source predictions, a weights learning method is developed to evaluate the contributions of sources. Many previous studies weigh source predictions using average mean method or by normalizing similarities based on distribution distance. A simple and usual similarity learning is

$$\omega_{\text{sim}}^{k} = \frac{1}{\text{Dis}(\boldsymbol{X}_{s_k}, \boldsymbol{X}_t)}, \qquad \omega_{s_k} = \frac{\omega_{\text{sim}}^{k}}{\sum_{k'=1}^{K}\omega_{\text{sim}}^{k'}} \quad (14)$$

$\text{Dis}(\boldsymbol{X}_{s_k}, \boldsymbol{X}_t)$ means the distribution distance of kth source and target domains in shared feature space. In our work, it is

$$\text{Dis}(\boldsymbol{X}_{s_k}, \boldsymbol{X}_t) = \mathcal{MMD}(\boldsymbol{X}_{s_k}, \boldsymbol{X}_t) = \left\| \frac{1}{n_{s_k}}\sum_{i=1}^{n_{s_k}}\mathcal{K}\left(\boldsymbol{f}_{s_k}^{i}\right) - \frac{1}{n_t}\sum_{j=1}^{n_t}\mathcal{K}\left(\boldsymbol{f}_{t}^{j}\right)\right\|_{\mathcal{H}}^{2}. \quad (15)$$

The corresponding target prediction can be expressed as:

$$\boldsymbol{Y}_t = \sum_{k=1}^{K}\omega_{s_k}\cdot P_{s_k}\left(\boldsymbol{F}_{\text{cat}_{t_k}}\right). \quad (16)$$

This method indeed yields larger weights of the more similar source predictions. However, if the source performances have obvious differences, the minor disparities of weight values may fail to return preferable results on the target. To increase the disparities between source weights, we add an adjusting constant controlled by prediction labels to adjust the weight values. By doing this, the closest source domain is expected to dominate the prediction of target samples. As mentioned in (12), cross-domain constraint is used to ensure that the multiple source predictors could return the same label of the same target sample. Hence, weight adjustment mainly affects the target samples that are predicted differently by the source classifiers. For those samples, strengthening the importance of the target labels returned by the closest source domain and weakening those returned by the furthest source domains could guarantee that we get the correct labels with high probability.

It is assumed that the same predicted pseudo target labels returned by source predictors are "correct labels." These "correct labels" are used to decide when the original source weights ω_{s_k} should be adjusted. Since the prediction learning is processed based on batches and not the whole data set, the threshold value of "correct labels" is set as a, while the number of target samples in every iteration is b. The threshold means the source classifiers perform quite stably on the target domain, indicating that there is no need to adjust weights over each sample, which might be time consuming.

Pseudo labels returned by each source predictor is

$$\boldsymbol{Y}_{t_k} = P_{s_k}(\boldsymbol{F}_{\text{cat}_{t_k}}). \quad (17)$$

Then the "correct labels" can be expressed as:

$$\boldsymbol{Y}_{tc} = \mathcal{Z}\left(\{\boldsymbol{Y}_{t_k}\}_{k=1}^{K}\right), \qquad n_{tc} = \mathcal{C}(\boldsymbol{Y}_{tc}) \quad (18)$$

where $\mathcal{Z}$ is the operation to get the same predicted labels, $\mathcal{C}$ means function to count the number n_{tc} of "correct labels." When $n_{tc} >= a$, the source weights in (14) can be rewritten as

$$\begin{aligned}
\omega_{sk} &= R(G(\omega_{s_k} + (1 - a/b))) \\
\omega_{s_k} &= \omega_{sk} + \frac{a}{K \cdot b} + \frac{(K-2) \cdot a}{K(K-1) \cdot b} \\
\text{if. } \omega_{s_k} &= \max[\omega_{s_1}, \omega_{s_2}, \ldots, \omega_{s_K}] \\
\omega_{sk} &= R(G(\omega_{s_k} - (1 - a/b))) \\
\omega_{s_k} &= \omega_{sk} - \frac{a}{K \cdot b} \\
\text{if. } \omega_{s_k} &= \min[\omega_{s_1}, \omega_{s_2}, \ldots, \omega_{s_K}] \\
\omega_{sk} &= R(G(\omega_{s_k})) \\
\omega_{s_k} &= \omega_{sk} - \frac{a}{K(K-1) \cdot b}
\end{aligned} \tag{19}$$

where G is sigmoid function, R is normalized function $R = (\omega_{sk}/(\sum_{sk'=1}^{K} \omega_{sk'}))$, ω_{s_k} satisfy $\sum_{k=1}^{K} \omega_{s_k} = 1$. Apply the above new weights to equation (16) when n_{tc} is larger than threshold, target labels can be predicted.

The whole processing is described in Algorithm 1.

Algorithm 1 Proposed Multi-source Domain Adaptation Method

Input: Source domains $\{\mathcal{D}_{s_k}\}_{k=1}^{K}$, target domain $\mathcal{D}_t$, training iteration $\mathcal{I}$, pretrained model $\phi(\cdot, \boldsymbol{\theta})$;
Initialization: Feature extraction networks $\{\phi_{c_k}(\cdot, \boldsymbol{\theta}_{c_k})\}_{k=1}^{K}$, $\{\phi_{d_k}(\cdot, \boldsymbol{\theta}_{d_k})\}_{k=1}^{K}$, and source predictors $\{P_{s_k}\}_{k=1}^{K}$;
for $\epsilon = 1, \epsilon < \mathcal{I}, \epsilon++,$ **do**
  $\{(\boldsymbol{x}_{s_k}, \boldsymbol{y}_{s_k})\}_{k=1}^{K} \leftarrow$ collect m batch pairs from corresponding $\mathcal{D}_{s_k}$ randomly;
  $\{\boldsymbol{x}_t\} \leftarrow$ collect m batch pairs from $\mathcal{D}_t$ randomly;
  $\{\boldsymbol{f}_{s_k}, \boldsymbol{f}_t\}_{k=1}^{K} \leftarrow \{\phi(\{\boldsymbol{x}_{s_k}, \boldsymbol{x}_t\}, \boldsymbol{\theta})\}_{k=1}^{K}$, collect shared features according to (1);
  $\{\boldsymbol{f}_{c_{s_k}}, \boldsymbol{f}_{c_{t_k}}\}_{k=1}^{K} \leftarrow \{\phi_{c_k}(\{\boldsymbol{f}_{s_k}, \boldsymbol{f}_t\}, \boldsymbol{\theta}_{c_k})\}_{k=1}^{K}$, collect common features according to (2);
  $\{\boldsymbol{f}_{d_{s_k}}, \boldsymbol{f}_{d_{t_k}}\}_{k=1}^{K} \leftarrow \{\phi_{d_k}(\{\boldsymbol{f}_{s_k}, \boldsymbol{f}_t\}, \boldsymbol{\theta}_{d_k})\}_{k=1}^{K}$, collect diverse features according to (3);
  $\mathcal{L}_s \leftarrow \mathcal{L}_c - \mathcal{L}_d$, compute loss within source domains according to (4), (5) and (6);
  $\mathcal{L}_t \leftarrow \mathcal{L}_{domain} + \mathcal{L}_{class}$, compute loss between source and target domains according to (7), (9), and (10);
  Compute prediction loss $\mathcal{L}_p$ according to (11);
  Compute cross-domain constrain loss $\mathcal{L}_{cro}$ according to (12);
  Compute total loss $\mathcal{L}$ according to (13);
  Compute ω_{s_k} according to (14);
  $\{\boldsymbol{Y}_{t_k}\}_{k=1}^{K} \leftarrow \{P_{s_k}(\boldsymbol{F}_{cat_{t_k}})\}_{k=1}^{K}$, collect pseudo labels according to (17);
  $\boldsymbol{Y}_{tc} \leftarrow \mathcal{Z}(\{\boldsymbol{Y}_{t_k}\}_{k=1}^{K})$, $n_{tc} \leftarrow \mathcal{C}(\boldsymbol{Y}_{tc})$, collect the same labels according to (18);
  if $n_{tc} >= a$ **then**
    Adjust ω_{s_k} according to (19);
  end if
  $\boldsymbol{Y}_t \leftarrow \sum_{k=1}^{K} \omega_{s_k} \cdot P_{s_k}(\boldsymbol{F}_{cat_{t_k}})$, return target labels according to (16)
  Update $\phi(\cdot, \boldsymbol{\theta})$, $\{\phi_{c_k}(\cdot, \boldsymbol{\theta}_{c_k})\}_{k=1}^{K}$, $\{\phi_{d_k}(\cdot, \boldsymbol{\theta}_{d_k})\}_{k=1}^{K}$, and source predictors $\{P_{s_k}\}_{k=1}^{K}$;
end for
Output: Predicted target label $\boldsymbol{Y}_t$.

IV. Experiments

In this section, we apply the proposed method to some popular real-world visual data sets for multiple sources domain adaptation classification tasks. Results, comparison, and analysis will be provided.

A. Data Sets and Baselines

Experimental data sets include Office-31, ImageCLEF-DA, Office-Home, and Office-Caltech10.

Office-31 is an unbalanced data set comprising 4110 images from data sets Amazon (A), Webcam (W), and DSLR (D) which share 31 categories, and each data set is regarded as a domain. Amazon contains 2817 images, Webcam has 795 images, and DSLR holds 498 images. The number of images in each category is different. Proposed method is tested via building three tasks: $A, W \rightarrow D$; $A, D \rightarrow W$; and $D, W \rightarrow A$.

ImageCLEF-DA is a balanced data set containing 1800 images from data sets Caltech-256 (C), ImageNet ILSVRC 2012 (I), and Pascal VOC 2012 (P) which share 12 categories, each domain corresponding to a data set. Every category contains 50 images and there are 600 images in each domain. Proposed method is tested via building three tasks: $I, C \rightarrow P$; $I, P \rightarrow C$; and $C, P \rightarrow I$.

Office-Home is a new and large unbalanced data set consisting of 15588 images from data sets Art (A), Clipart (C), Product (P), and Real World (R) which share 65 categories. Art has 2427 images, Clipart contains 4365 images, Product comprises 4439 images, and Real World holds 4357 images. Treating each data set as a domain, the proposed method is tested via building four tasks: $A, C, P \rightarrow R$; $A, C, R \rightarrow P$; $A, P, R \rightarrow C$; and $C, P, R \rightarrow A$.

Office-Caltech10 is an unbalanced data set extended by Office-31 and Caltech, which consists 2533 images sharing 10 categories. Caltech (C) contains 1123 images, Amazon (A) contains 958 images, Webcam (W) holds 295 images, and DSLR (D) has 157 images. Treating each data set as a domain, proposed method is tested via building four tasks: $A, D, W \rightarrow C$; $C, D, W \rightarrow A$; $A, C, D \rightarrow W$; and $A, C, W \rightarrow D$.

There are three standards: "single best," "source combine," and "multi-source." "Single best" means the best performance of single-source domain using the single-source domain adaptation method, "source combine" is performance returned by a single-source domain adaptation method with multiple sources, which unites all source domains as one, "multi-source" is domain adaptation with multiple sources, all methods complete target task using different combination rules. Comparable state-of-the-art domain adaptation methods are as follows. The single-source domain adaptation methods include:

1) DAN: Deep adaptation network [15];
2) RevGrad: Reverse gradient [16];

Fig. 4. Training loss of the proposed method. Taking task Amazon in Office-31 as an example.

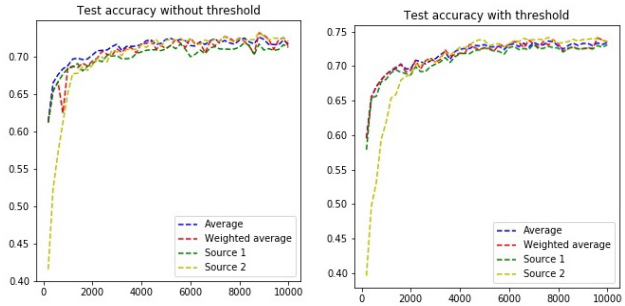


Fig. 5. Test accuracy of the proposed method without and with using threshold. Taking task Amazon in Office-31 as an example. (a) Without threshold. (b) With threshold.

3) D-CORAL: Correlation alignment for domain adaptation [17];
4) MRAN: Multi-representation adaptation network [18];
5) MADA: Multi-adversarial domain adaptation [40];
6) DAAN: Dynamic adversarial adaptation network [41];
7) ADDA: Adversarial discriminative domain adaptation [56];
8) CyCADA: Cycle-consistent adversarial domain adaptation [57].

The multi-source domain adaptation methods include:

1) DCTN: Deep cocktail network [28];
2) M3SDA: Moment matching for multi-source domain adaptation [29];
3) MFSAN: Multiple feature spaces adaptation network [26].

All results for comparison are collected from previous studies based on ResNet, except for MFSAN on data sets Office-Home and Office-Caltech10, and we ran them ourselves using code released by authors.[1]

B. Parameter Setting and Effect of Different Similarity Metrics

Our experiments were performed using Pytorch based on ResNet50 (shared network). Feature extraction networks $\phi_{c_k}(\cdot, \boldsymbol{\theta}_{c_k})$ and $\phi_{d_k}(\cdot, \boldsymbol{\theta}_{d_k})$ have three convolution layers, the source predictors P_{s_k} contains one fully connected layer. We fine-tune all convolutional layers using back-propagation with stochastic gradient descent (SGD), the momentum is 0.9, the learning rate η follows the same strategy in [16], that is $\eta = (\eta_0/((1 + 10p)^{0.75}))$, where $\eta_0 = 0.01$, p is the training progress changing linearly from 0 to 1. Learning rate of shared network is one tenth of other layers. Batchsize $b = 32$, trade-off parameters $\lambda = 0.01$, α, β, γ follow existing work [26], that is $\alpha = \beta = \gamma = (2/(1 + \exp(-10p'))) - 1$, where p' changes from 0 to 1 linearly.

Threshold value a is defined by target task and varies accordingly. This also determines when the adjustment of weights should be started. We will explain how to choose an appropriate threshold value. Fig. 4 shows the source training losses of one experiment using the proposed method, and Fig. 5 displays the test accuracy of experiments with and without using threshold.

[1]https://github.com/easezyc/deep-transfer-learning

It can be seen that early of the training process (below 2000 times), the training loss of each source reduces sharply. Combining with Fig. 5, the test accuracy on task increases markedly. If we adjust all weights without the control of threshold, the accuracy of multi-source domain adaptation is near to single-source performance from the very beginning of training. But at that time, all predictions of single-source domain is not yet convergent, which means they cannot perform well on the target domain, thus giving a very large weight of a source which can harm the performance. It is appropriate to adjust weights when the training losses of sources start to reduce slowly. Changing weights by observing loss reduction during experiments might be inconvenient to operate, so we turn to observe the same target labels returned by source domains.

Fig. 6 shows the same target labels returned by source predictors in every batch. As training progresses, the number of the same target labels n_{tc} increases. For small data sets, this number falls largely in the interval between 29 and 32, for large data set, the value fluctuates mainly between 18 and 22, for data sets with a medium number of samples, the value falls between 27 and 30. The most frequent number means the performance of single source starts to become stable. In other words, adjustment of weights should be started before the occurrence of these values. A small threshold value means that the adjustment of weights starts early, while a large one means that in most cases, there is no need to adjust the weights. Normally, the larger the n_{tc} is, the less will be the differences among single-source predictions.

Fig. 7 shows test accuracy with different threshold values. The performance of final target predictor changes with different thresholds. If the threshold is too large compared with the most frequently occurring number of the same labels (rarely adjust weights), the accuracy is reduced. When dealing with a large quantity of experiments if we find the threshold of each different task, for convenience, instead of learning specific a for each target domain in every data set, we set $a = 30$ for small data sets ImageCLEF-DA and Office-Caltech10, $a = 24$ for data set Office-31, and $a = 20$ for large data set Office-Home.

In this article, we choose MMD as a similarity metric to measure the distance between two distributions. To evaluate

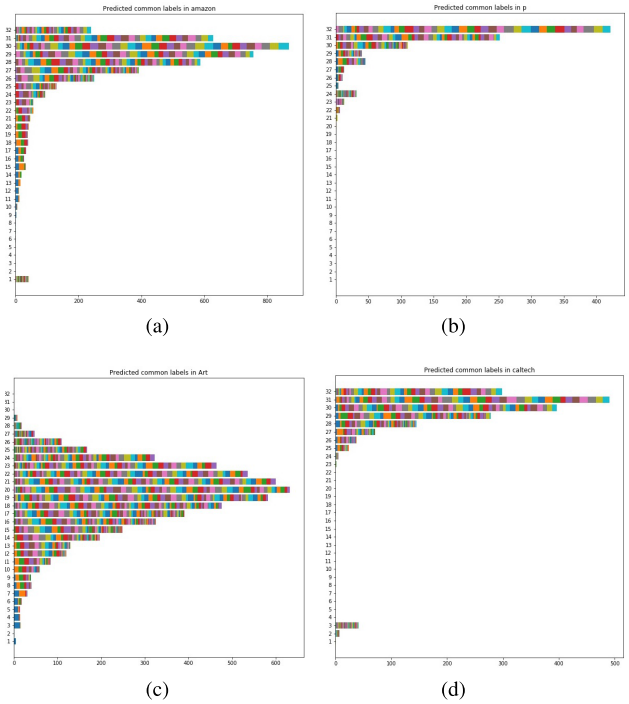


Fig. 6. Number of same labels returned by source predictors. Taking target domain Amazon as example for Office-31, Pascal VOC 2012 for ImageCLEF-DA, Art for Office-Home and Caltech for Office-Caltech10. (a) Office-31. (b) ImageCLEF-DA. (c) Office-Home. (d) Office-Caltech10.

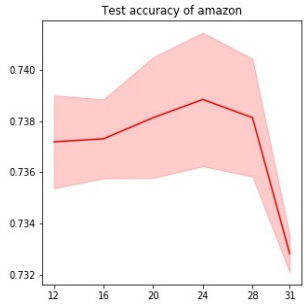


Fig. 7. Test accuracy of the proposed method with different threshold values. Taking task Amazon in Office-31 as an example, the accuracy is average result of three times experiments. The red line represents accuracy while the light red area signifies standard deviation.

the effectiveness of MMD, experiments based on another popular discrepancy measurement named Wasserstein distance (WD) are taken as a comparison on data set Office-31. The source order is the same as described, for example, $S1$ is domain A while $S2$ is domain W in task $A, W \rightarrow D$. "S" means single source and "M" means multi-source. Experiments are repeated for three times. Table I indicates that the model trained with MMD outperforms the model trained with WD, it achieves higher accuracy of both single-source and multi-source domain adaptation.

C. Results and Analysis of Proposed and Comparison With State-of-the-Art

For each data set, we run the proposed method five times with random initialized parameters and return the average performance. Tables II–V show results of the proposed and compared methods on Office-31, ImageCLEF-DA, Office-Home, and Office-Caltech10, respectively. It can be seen that the proposed method outperforms other state-of-the-art domain adaptation methods, and obtains the highest accuracy on most target tasks.

In general, domain adaptation with multiple source domains shows superior results compared with single best results. That means multiple sources with richer transferable information have positive influence on target task. At the same time, multi-source domain adaptation with combination rules performs better than simply combining all source domains as one.

TABLE I
ACCURACY (%) WITH DIFFERENT SIMILARITY METRICS

Standards		A, W-D	A, D-W	W, D-A	Avg
WD	S1	96.1	98.0	71.6	89.4
	S2	99.7	98.6	72.4	
	M	99.7	98.6	72.2	90.2
MMD	S1	96.8	98.1	73.3	90.1
	S2	99.7	98.8	**74.0**	
	M	**99.8**	**98.9**	73.9	**90.9**

TABLE II
COMPARISON OF CLASSIFICATION ACCURACY (%) ON DATA SET OFFICE-31

Standards	Method	A, W-D	A, D-W	W, D-A	Avg
Single best	ResNet	99.3	96.7	62.5	86.2
	DAN	99.5	96.8	66.7	87.7
	D-CORAL	99.7	98.0	65.3	87.7
	RevGard	99.1	96.9	68.2	88.1
	MADA	99.6	97.4	70.3	89.1
	MRAN	**99.8**	96.9	70.9	89.2
Source Combine	DAN	99.6	97.8	67.6	88.3
	D-CORAL	99.3	98.0	67.1	88.1
	RevGard	99.7	98.1	67.6	88.5
Multi-Source	DCTN	99.3	98.2	64.2	87.2
	MFSAN	99.5	98.5	72.7	90.2
	MSCLDA	**99.8**	**98.8**	**73.7**	**90.8**

TABLE III
COMPARISON OF CLASSIFICATION ACCURACY (%) ON DATA SET IMAGECLEF-DA

Standards	Method	I, C-P	I, P-C	P, C-I	Avg
Single best	ResNet	74.8	91.5	83.9	83.4
	DAN	75.0	93.3	86.2	84.8
	D-CORAL	76.9	93.6	88.5	86.3
	RevGard	75.0	**96.2**	87.0	86.1
	DAAN	78.5	94.3	91.3	88.0
	MADA	75.2	96.0	88.8	86.7
	MRAN	78.8	95.0	93.5	89.1
Source Combine	DAN	77.6	93.3	92.2	87.7
	D-CORAL	77.1	93.6	91.7	87.5
	RevGard	77.9	93.7	91.8	87.8
Multi-Source	DCTN	75.0	95.7	90.3	87.0
	MFSAN	79.1	95.4	93.6	89.4
	MSCLDA	**79.5**	95.9	**94.3**	**89.9**

TABLE IV
COMPARISON OF CLASSIFICATION ACCURACY (%) ON DATA SET OFFICE-HOME

Standards	Method	A,C,P-R	A,C,R-P	A,P,R-C	C,P,R,-A	Avg
Single best	ResNet	75.4	79.7	49.6	65.3	67.5
	DAN	75.9	80.3	56.5	68.2	70.2
	D-CORAL	76.3	80.3	53.6	67.0	69.3
	RevGard	75.8	80.4	55.9	67.9	70.0
	DAAN	74.0	78.8	54.0	66.3	68.3
	MRAN	77.5	**82.2**	60.0	70.4	72.5
Source Combine	DAN	82.5	79.0	59.4	68.5	72.4
	D-CORAL	**82.7**	79.5	58.6	68.1	72.2
	RevGard	**82.7**	79.5	59.1	68.4	72.4
Multi-Source	MFSAN	80.8	79.0	60.7	70.0	72.6
	MSCLDA	80.6	79.9	**61.4**	**71.6**	**73.4**

TABLE V
COMPARISON OF CLASSIFICATION ACCURACY (%) ON DATA SET OFFICE-CALTECH10

Standards	Method	A,D,W-C	C,D,W-A	A,C,D-W	A,C,W-D	Avg
Single best	ResNet	82.5	91.2	98.9	99.2	93.0
	ADDA	88.8	94.5	99.1	98.0	95.1
	CyCADA	89.7	96.2	98.9	97.3	95.5
Source Combine	DAN	89.7	94.8	99.3	98.2	95.5
	ADDA	90.2	95.0	99.4	98.2	95.7
	CyCADA	91.0	95.9	99.0	97.8	95.9
Multi-Source	DCTN	90.2	92.7	99.4	99.0	95.3
	M3SDA	92.2	94.5	**99.5**	98.2	96.4
	MFSAN	93.8	95.1	99.1	**98.7**	96.7
	MSCLDA	**94.1**	**95.3**	99.1	98.5	**96.8**

Simply combining them fails to consider the specific knowledge contained in each source domain, and transforms features of all domains into a common latent feature space. However, sometimes, a feature space that can adapt all domain distributions may not exist, which means the predictions learned based on these features in that same space may not work as well on both source and target domains as desired. Multi-source domain adaptation with combination rules, on the contrary, explores common features as well as specific features, and learn specific predictors of source domains, by which the distributions of source and target domains can be better matched.

Tables VI–IX present classification accuracy on Office-31, ImageCLEF-DA, Office-Home, and Office-Caltech10 with different combination rules and different distribution matching strategies. Since the results of averaged combination and of weighted combination without adjustment we observed show less difference (the details will be provided below in Fig. 8), we only compared results of mean method and weighted method with adjustment. "Domain-level only" means the proposed method without class-level distribution matching, "multi-level" means the proposed method with domain-level and class-level distributions matching. The source orders are the same as the task name. For example, in task $A, D \rightarrow W$, $S1$ is domain A, and $S2$ is domain D. Average result of $S1$ and $S2$ represents average accuracy of all single domain adaptation tasks. "Sbest" means the best performance of single-source domain using the proposed method. "MeanC" is the proposed multi-source domain adaptation method using the averaged combination rule.

TABLE VI
COMPARISON OF CLASSIFICATION ACCURACY (%) ON DATA SET OFFICE-31 WITH DIFFERENT COMBINATION RULES

Standards	Method	A, W-D	A, D-W	W, D-A	Avg
Domain-level only	S1	97.5	97.2	71.3	89.4
	S2	99.6	98.6	72.3	
	Sbest	99.6	98.6	72.3	90.2
	MeanC	99.1	98.3	71.8	89.7
	MSCLDA	99.6	98.7	72.1	90.1
Multi-level	S1	96.7	98.0	72.9	89.9
	S2	99.7	98.7	**73.8**	
	Sbest	99.7	98.7	**73.8**	90.7
	MeanC	98.7	98.7	73.3	90.3
	MSCLDA	**99.8**	**98.8**	73.7	**90.8**

TABLE VII
COMPARISON OF CLASSIFICATION ACCURACY (%) ON DATA SET IMAGECLEF-DA WITH DIFFERENT COMBINATION RULES

Standards	Method	I, C-P	I, P-C	P, C-I	Avg
Domain-level only	S1	79.0	95.7	93.1	89.3
	S2	79.0	95.7	93.2	
	Sbest	79.0	95.7	93.2	89.3
	MeanC	79.1	95.8	93.0	89.3
	MSCLDA	79.0	**95.9**	93.2	89.4
Multi-level	S1	79.4	95.7	94.0	89.7
	S2	79.4	95.6	94.2	
	Sbest	79.4	95.7	94.2	89.8
	MeanC	**79.6**	95.7	**94.4**	**89.9**
	MSCLDA	79.5	**95.9**	94.3	**89.9**

TABLE VIII
COMPARISON OF CLASSIFICATION ACCURACY (%) ON DATA SET OFFICE-HOME WITH DIFFERENT COMBINATION RULES

Standards	Method	A,C,P-R	A,C,R-P	A,P,R-C	C,P,R,-A	Avg
Domain-level only	S1	76.4	72.3	57.6	64.5	68.9
	S2	75.2	73.3	56.7	65.6	
	S3	78.2	78.3	59.4	69.8	
	Sbest	78.2	78.3	59.4	69.8	71.4
	MeanC	80.3	77.8	60.9	69.5	72.1
	MSCLDA	**80.6**	78.8	61.1	70.0	72.6
Multi-level	S1	78.1	73.3	57.8	66.3	71.0
	S2	77.6	77.4	59.5	67.4	
	S3	80.0	**80.3**	61.3	**72.4**	
	Sbest	80.0	**80.3**	61.3	**72.4**	**73.5**
	MeanC	80.4	78.4	61.2	69.7	72.4
	MSCLDA	**80.6**	79.9	**61.4**	71.6	73.4

In most cases, multi-source domain adaptation outperforms single-source domain adaptation. Predictions with multi-level distribution matching return higher accuracy than those without matching class-level distribution. Multi-source domain adaptation with weight adjustment outperforms the results without weight adjustment.

For small data sets ImageCLEF-DA and Office-Caltech10, classification accuracy of the proposed method has little difference to that of the average mean method. This may because the performance of single-source domain adaptation is fairly similar to other single-source domains. Fig. 6 in Section IV-B also indicates that the number of the same target labels returned by all single-source predictors is near to batch size,

TABLE IX

COMPARISON OF CLASSIFICATION ACCURACY (%) ON DATA SET OFFICE-CALTECH10 WITH DIFFERENT COMBINATION RULES

Standards	Method	A,D,W-C	C,D,W-A	A,C,D-W	A,C,W-D	Avg
Domain-level only	S1	92.5	94.8	97.5	96.7	95.8
	S2	92.4	94.4	97.3	98.1	
	S3	93.4	94.3	98.8	99.7	
	Sbest	93.4	94.8	98.8	99.7	96.7
	MeanC	93.7	**95.4**	98.8	98.1	96.5
	MSCLDA	93.7	95.3	98.8	98.5	96.6
Multi-level	S1	93.4	94.8	98.4	96.7	96.2
	S2	93.6	94.4	98.6	96.7	
	S3	**94.1**	94.9	**99.2**	**100.0**	
	Sbest	**94.1**	94.9	**99.2**	**100.0**	**97.1**
	MeanC	**94.1**	**95.4**	99.1	98.5	96.8
	MSCLDA	**94.1**	95.3	99.1	98.5	96.8

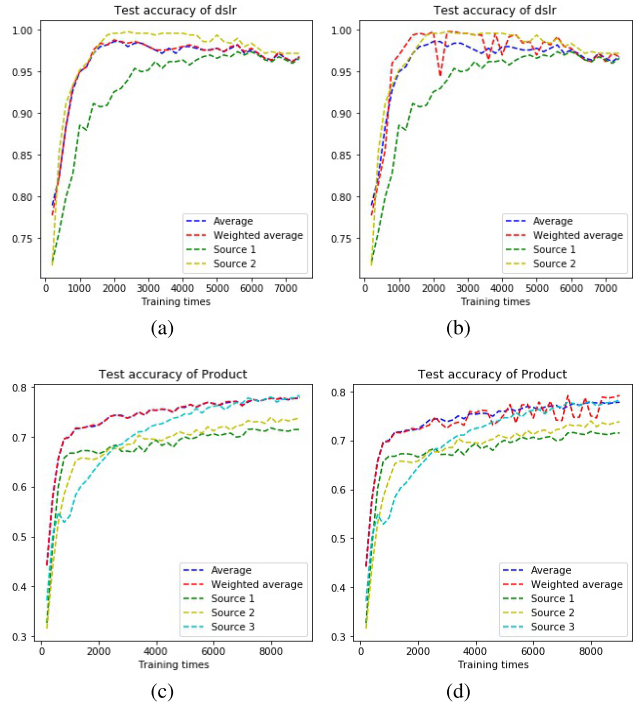


Fig. 8. Classification accuracy without and with adjusting weights. Figures (a) and (c) are results without adjustment, while figures (b) and (d) are results with adjustment. (a) Without adjustment. (b) With adjustment. (c) Without adjustment. (d) With adjustment.

most often being 31 or 32. This represents that here, average mean combination can achieve almost the same performance as weighted average mean combination. It also can be seen that the single best performance of the proposed method is better than single best results provided in Tables II–V, which means the cross-domain constraint can improve the transferability of single-source domain.

To detail the impacts of weights, taking DSLR (target domain) in data set Office-31 as an example for two-source domain adaptation, and Product in data set Office-Home as an example for three-source domain adaptation, Fig. 8 indicates the results of classification accuracy with and without adjustment.

It can be seen that the accuracy of average mean method and that of weighted mean method without adjustment has no significant difference. The line of average combination is

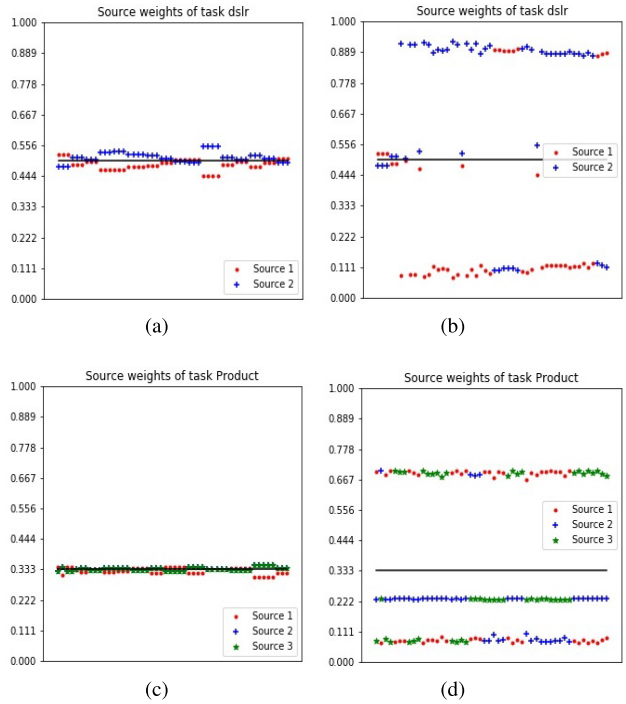


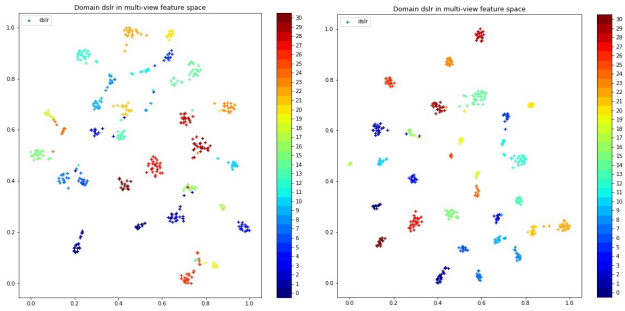
Fig. 9. Source weights without and with adjusting. Figures (a) and (c) are results without adjustment, while figures (b) and (d) are results with adjustment. (a) Without adjustment. (b) With adjustment. (c) Without adjustment. (d) With adjustment.

almost superimposed on that of weighted combination without adjustment. If the single-source domain adaptation has obvious disparity, the mean combination or weighted combination without adjustment cannot take the advantage of the single best one and return preferable results. The proposed method, however, is superior to them in these cases.

The line of the proposed method displays some fluctuations that could result from wrong original weights, which means the well-performing source domain is given small weight while the inferior ones get large weights. We may take this as future work to explore how to learn more reliable weights that are concordant with their performance on the target domain.

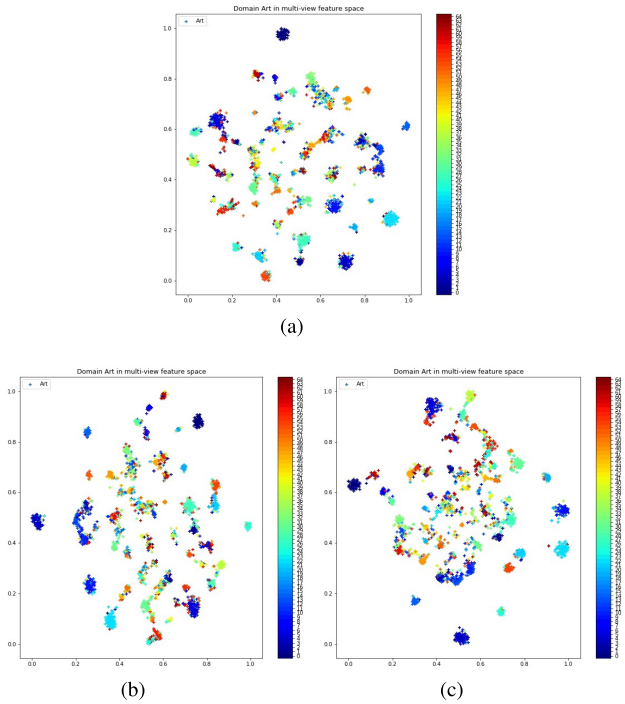
Fig. 9 shows weights with and without adjustment. Since the adjustment is based on batches, there are too many weights during one experiment. So we randomly choose 50 values of each task to draw the pictures. Taking DSLR (target domain) in data set Office-31 as an example for two-source domain adaptation, and Product in data set Office-Home as an examples for three-source domain adaptation, there is some evidence as to why the results of the average mean method is almost the same as the results of weighted mean method without adjustment.

The picture shows that all weights without adjustment are around the mean value of the greater frequencies. Thus, their results rarely differ markedly from each other, while the adjusted weights show significant disparity. For two-source domain adaptation, weights are near to 0 and 1, for three-source domain adaptation, only the smallest weights are near to boundary. It might need further exploration if the largest weights require extra adjustment to make it more closer to 1.



(a) (b)

Fig. 10. T-SNE visualization of target with different source domain, take task DSLR in Office-31 as example. (a) A-D. (b) W-D.



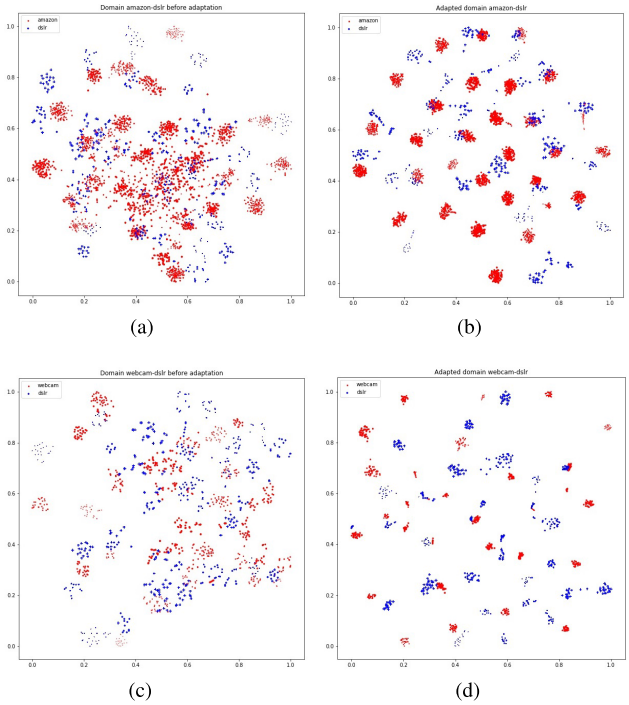
(a) (b) (c)

Fig. 11. T-SNE visualization of target with different source domain, take task Art in Office-Home as example. (a) C-A. (b) P-A. (c) R-A.

D. Visualization Analysis of Proposed

To show the efficiency of domain adaptation using the proposed method, this section displays the visualization of source and target features, which transforms high dimension data into 2-D space to display the domain categories directly. Figs. 10 and 11 show t-SNE visualization [58] of classification features in target domain with different single-source domain. Let Office-31 represents two-source domain adaptation and Office-Home represents three-source domain adaptation. For data set with a small number of categories, it shows that each category separates clearly from others, while task W-D discriminates the categories more clearly than task A-D. This is concordant with classification accuracy shown in Table VI, source domain Webcam returns the single best results.

For very large data set with more categories, only partial of the boundaries between each two different categories can be discriminated clearly, while the remaining categories may



(a) (b) (c) (d)

Fig. 12. T-SNE visualization of features before and after adaptation, taking task DSLR in Office-31 as example for two-source domain adaptation. (a) A-D before adaptation. (b) A-D after adaptation. (c) W-D before adaptation. (d) W-D after adaptation.

seem too close to each other. Combining results provided in Tables IV and VIII, the target domain classification accuracy of the whole data set using different methods falls to a fairly low level compared with other data sets (of which the average accuracy is commonly around 90%). So, it is reasonable that the distance between different categories is not as great as that of data sets with small categories.

To display effects of distribution matching, Figs. 12 and 13 show t-SNE visualization of domain features in shared feature space (before adaptation) and multi-view feature space (after adaptation). The red indicates source domain, while blue shows target domain.

It can be seen how class distribution in multi-view feature space shows them clearly separated from each other while that in shared feature space has misalignments. For all source domains, whether the categories and samples are small or large, the distance between each two classes is sufficient and without any superposition. For target domains, adaptation is achieved well for data set with small categories, each category in the target domain is mapped with that in the source domain by a short distance. While for Office-Home with large numbers of samples and categories, as mentioned before, the distance among classes after adaptation may not seem as clear as with a small data set, but, compared with that in shared feature space, the distribution matching still splits the different categories.

E. Ablation Study and Sample Complexity

An ablation study based on data set Office-31 is performed to show the effectiveness of the loss components. The constraints of source domain adaptation (L_s), target domain

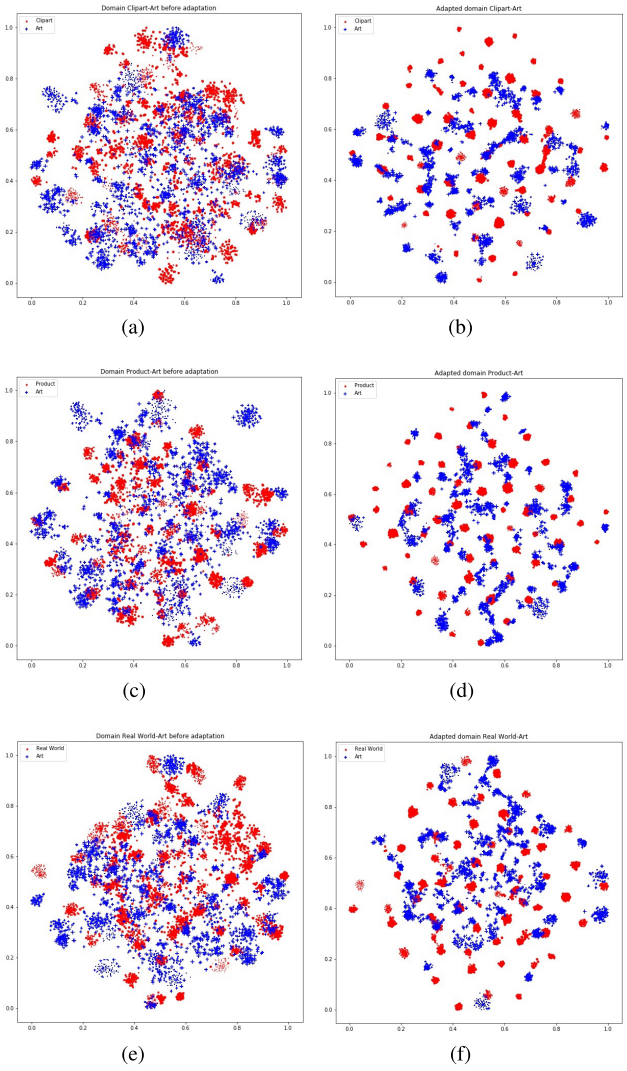


Fig. 13. T-SNE visualization of features before and after adaptation, taking task Art in Office-Home as example for three-source domain adaptation. (a) C-A before adaptation. (b) C-A after adaptation. (c) P-A before adaptation. (d) P-A after adaptation. (e) R-A before adaptation. (f) R-A after adaptation.

adaptation on domain-level (L_{domain}), target domain adaptation on class-level (L_{class}) and cross-domain alignment (L_{cro}) are regarded as the control variables, and each of them is removed in turn to show its contribution of the learning performance. All experiments are repeated for three times, and the results are shown in Table X.

It can be seen that the target domain adaptation on domain-level and the cross-domain alignment contribute more than the other constraints, because the classifier trained without either of them returns the lowest accuracy. Domain adaptation on class-level plays an auxiliary role of domain-level adaptation to help improve the performance. Source domain adaptation shows superiority in the target task which contains a large number of samples.

To show the influence of training sample size on the learning performance, sample complexity experiments are taken on data set Office-31. For each task, we randomly select 25%, 50%, and 75% source samples to train the classifier and compare its performance with the classifier that is trained using all source samples. The results are shown in Table XI.

TABLE X

ACCURACY (%) WITHOUT DIFFERENT LOSS COMPONENTS

Standards	A, W-D	A, D-W	W, D-A	Avg
Without L_s	**99.9**	**98.9**	73.0	90.6
Without L_{domain}	99.7	98.1	71.4	89.7
Without L_{class}	99.7	**98.9**	72.5	90.4
Without L_{cro}	99.8	97.9	71.0	89.6
Proposed	99.8	**98.9**	**73.9**	**90.9**

TABLE XI

ACCURACY (%) WITH DIFFERENT TRAINING SAMPLE SIZE

Standards	A, W-D	A, D-W	W, D-A	Avg
25% samples	95.6	96.3	67.1	86.3
50% samples	97.5	98.2	71.7	89.1
75% samples	98.7	98.1	73.4	90.1
100% samples	**99.8**	**98.9**	**73.9**	**90.9**

It can be seen that with the growth of the training sample size, the performance improves. The greatest increase occurs at when the sample size increases from 25% to 50%, after which the growth slows. For task $A, D \rightarrow W$, the performance of 50% and 75% samples is extremely close to each other. This indicates that when the pretrained networks are used as the backbone, the training sample quantity might not be the only main factor affecting the learning performance. Other factors such as the sample quality and domain similarity should also be taken into consideration.

V. CONCLUSION AND FURTHER STUDY

This section concludes the whole work and formulates the directions for further study.

In this article, we propose a source contributions learning method for multi-source domain adaption, where the multi-view feature extraction and multi-level distribution matching are employed to enhance transferability of domain adaptation. Compared with existing multi-source domain adaptation methods, ours not only explores the similarities among source and target domains, but also learns diversities of a target domain and turns it into extracting multiple aspects of source domain features since the target data are unlabeled. At the same time, domain adaptation is achieved by adapting source domains to each other as well as adapting source and target domains using domain-level distribution matching and class-level distribution matching, which improve the classification accuracy by reducing the confusion of boundary samples. When it comes to completing a target task, instead of using the averaged combination rule or the traditional weighted combination rule, a weight adjustment strategy is developed based on pseudo target labels to increase the disparity of source weights, which can take advantage of the single best source domain when the performances of sources have significant differences. Experiments on real-world visual data sets evaluate the superiority of the proposed compared with other state-of-the-art domain adaptation methods using deep neural networks.

In the future, we might explore new methods to learn more reliable weights that can represent their performance on target domain exactly. By doing this, we expect to solve the problem where the source domain with poor accuracy on target attracts large weight. Another work is extending the proposed method to heterogeneous feature spaces. Features from different views may have their own best represented ability with different dimensions. This still needs further exploration.

REFERENCES

[1] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowl.-Based Syst.*, vol. 80, pp. 14–23, May 2015.

[2] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[3] L. Zhang and X. Gao, "Transfer adaptation learning: A decade survey," 2019, *arXiv:1903.04687*. [Online]. Available: http://arxiv.org/abs/1903.04687

[4] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[5] H. Zuo, G. Zhang, W. Pedrycz, and J. Lu, "Domain selection of transfer learning in fuzzy prediction models," in *Proc. IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE)*, Jun. 2019, pp. 1–6.

[6] J. Lu, H. Zuo, and G. Zhang, "Fuzzy multiple-source transfer learning," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 12, pp. 3418–3431, Dec. 2020.

[7] H. Yu, M. Hu, and S. Chen, "Multi-target unsupervised domain adaptation without exactly shared categories," 2018, *arXiv:1809.00852*. [Online]. Available: http://arxiv.org/abs/1809.00852

[8] Q. Wu *et al.*, "Online transfer learning with multiple homogeneous or heterogeneous sources," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 7, pp. 1494–1507, Jul. 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7883886

[9] K. Li, Y. Zhang, K. Li, Y. Li, and Y. Fu, "Attention bridging network for knowledge transfer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5198–5207.

[10] H. Zuo, J. Lu, G. Zhang, and W. Pedrycz, "Fuzzy rule-based domain adaptation in homogeneous and heterogeneous spaces," *IEEE Trans. Fuzzy Syst.*, vol. 27, no. 2, pp. 348–361, Feb. 2019.

[11] H. Li, S. J. Pan, R. Wan, and A. C. Kot, "Heterogeneous transfer learning via deep matrix completion with adversarial kernel embedding," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 8602–8609.

[12] F. Liu, G. Zhang, and J. Lu, "Heterogeneous domain adaptation: An unsupervised approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5588–5602, Dec. 2020.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[15] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 97–105.

[16] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1180–1189.

[17] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 443–450.

[18] Y. Zhu *et al.*, "Multi-representation adaptation network for cross-domain image classification," *Neural Netw.*, vol. 119, pp. 214–221, Nov. 2019.

[19] J. Zhang, W. Li, and P. Ogunbona, "Joint geometrical and statistical alignment for visual domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1859–1867.

[20] K. Saito, Y. Ushiku, and T. Harada, "Asymmetric tri-training for unsupervised domain adaptation," in *Proc. 34th Int. Conf. Mach. Learn.*, Vol. 70, 2017, pp. 2988–2997.

[21] Y. Wei, Y. Zhang, J. Huang, and Q. Yang, "Transfer learning via learning to transfer," in *Proc. Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 5085–5094.

[22] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation with multiple sources," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1041–1048.

[23] I. Redko, A. Habrard, and M. Sebban, "On the analysis of adaptability in multi-source domain adaptation," *Mach. Learn.*, vol. 108, nos. 8–9, pp. 1635–1652, Sep. 2019.

[24] Y. Li *et al.*, "Extracting relationships by multi-domain matching," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6798–6809.

[25] H. Zhao, S. Zhang, G. Wu, J. M. Moura, J. P. Costeira, and G. J. Gordon, "Adversarial multiple source domain adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 8559–8570.

[26] Y. Zhu, F. Zhuang, and D. Wang, "Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 5989–5996.

[27] F. Liu, G. Zhang, and J. Lu, "Multi-source heterogeneous unsupervised domain adaptation via fuzzy-relation neural networks," *IEEE Trans. Fuzzy Syst.*, early access, Aug. 20, 2020, doi: 10.1109/TFUZZ.2020.3018191.

[28] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3964–3973.

[29] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1406–1415.

[30] S. Zhao *et al.*, "Multi-source distilling domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12975–12983.

[31] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, Mar. 2012.

[32] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," 2014, *arXiv:1412.3474*. [Online]. Available: http://arxiv.org/abs/1412.3474

[33] A. Gretton *et al.*, "Optimal kernel choice for large-scale two-sample tests," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1205–1213.

[34] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2208–2217.

[35] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4893–4902.

[36] F. Liu, W. Xu, J. Lu, G. Zhang, A. Gretton, and D. J. Sutherland, "Learning deep kernels for non-parametric two-sample tests," 2020, *arXiv:2002.09116*. [Online]. Available: http://arxiv.org/abs/2002.09116

[37] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, "Central moment discrepancy (CMD) for domain-invariant representation learning," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–13.

[38] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.

[39] I. Redko, N. Courty, R. Flamary, and D. Tuia, "Optimal transport for multi-source domain adaptation under target shift," in *Proc. Int. Conf. Artif. Intell. Statist.*, vol. 89, 2019, pp. 849–858.

[40] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 1–8.

[41] C. Yu, J. Wang, Y. Chen, and M. Huang, "Transfer learning with dynamic adversarial adaptation network," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2019, pp. 778–786.

[42] D. Das and C. G. Lee, "Graph matching and pseudo-label guided deep unsupervised domain adaptation," in *Proc. Int. Conf. Artif. Neural Netw.* Cham, Switzerland: Springer, 2018, pp. 342–352.

[43] B. Yang and P. C. Yuen, "Cross-domain visual representations via unsupervised graph alignment," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 5613–5620.

[44] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, nos. 1–2, pp. 151–175, May 2010.

[45] G. Kang, L. Zheng, Y. Yan, and Y. Yang, "Deep adversarial attention alignment for unsupervised domain adaptation: The benefit of target expectation maximization," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 401–416.

[46] C. Lin, S. Zhao, L. Meng, and T.-S. Chua, "Multi-source domain adaptation for visual sentiment classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 2661–2668.

[47] J. Lee, P. Sattigeri, and G. Wornell, "Learning new tricks from old dogs: Multi-source transfer learning from pre-trained networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4372–4382.

[48] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation," 2020, *arXiv:2002.08546*. [Online]. Available: http://arxiv.org/abs/2002.08546
[49] X. Wang, L. Li, W. Ye, M. Long, and J. Wang, "Transferable attention for domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 5345–5352.
[50] J. Wen, R. Liu, N. Zheng, Q. Zheng, Z. Gong, and J. Yuan, "Exploiting local feature patterns for unsupervised domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 5401–5408.
[51] H. Li, S. J. Pan, S. Wang, and A. C. Kot, "Domain generalization with adversarial feature learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5400–5409.
[52] T. Matsuura and T. Harada, "Domain generalization using a mixture of multiple latent domains," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11749–11756.
[53] J. Zhang, W. Zhou, X. Chen, W. Yao, and L. Cao, "Multisource selective transfer framework in multiobjective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 24, no. 3, pp. 424–438, Jun. 2020.
[54] S. Zhao, B. Li, X. Yue, P. Xu, and K. Keutzer, "MADAN: Multi-source adversarial domain aggregation network for domain adaptation," 2020, *arXiv:2003.00820*. [Online]. Available: http://arxiv.org/abs/2003.00820
[55] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5018–5027.
[56] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7167–7176.
[57] J. Hoffman *et al.*, "CyCADA: Cycle-consistent adversarial domain adaptation," 2017, *arXiv:1711.03213*. [Online]. Available: http://arxiv.org/abs/1711.03213
[58] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

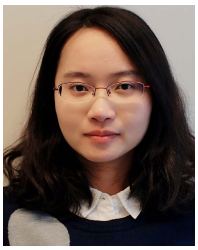
Keqiuyin Li is currently pursuing the Ph.D. degree with the Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia.

She is a member of the Decision Systems and eService Intelligence (DeSI) Research Laboratory, Centre for Artificial Intelligence (CAI), University of Technology Sydney. Her research interests include transfer learning and domain adaptation.

Jie Lu (Fellow, IEEE) received the Ph.D. degree from Curtin University, Perth, WA, Australia, in 2000.

She is currently a Distinguished Professor and the Director of the Australian Artificial Intelligence Institute (AAII), University of Technology Sydney (UTS), Sydney, NSW, Australia. She has also supervised 40 Ph.D. students to completion. Her main research expertise is in fuzzy transfer learning, concept drift, decision support systems, and recommender systems. She was awarded with ten Australian Research Council (ARC) discovery grants and led 15 industry projects.

Dr. Lu is a fellow of the IFSA and also an Australian Laureate Fellow. She has received the UTS Medal for Research and Teaching Integration in 2010, *The Computer Journal* Wilkes Award in 2018, the UTS Medal for Research Excellence in 2019, the IEEE TRANSACTIONS ON FUZZY SYSTEMS Outstanding Paper Award in 2019, and the Australian Most Innovative Engineer Award in 2019. She serves as the Editor-in-Chief for *Knowledge-Based Systems* (Elsevier) and the *International Journal on Computational Intelligence Systems* (Atlantis). She has delivered 27 keynote speeches at IEEE and other international conferences and has chaired 15 international conferences.



Hua Zuo (Member, IEEE) received the Ph.D. degree from the University of Technology Sydney, Sydney, NSW, Australia, in 2018.

She is currently a Lecturer with the Faculty of Engineering and Information Technology, University of Technology Sydney, where she is also a member of the Decision Systems and e-Service Intelligence (DeSI) Research Laboratory, Centre for Artificial Intelligence. Her research interests include transfer learning and fuzzy systems.

Guangquan Zhang received the Ph.D. degree in applied mathematics from Curtin University, Perth, WA, Australia, in 2001.

He is currently an Associate Professor and the Director of the Decision Systems and e-Service Intelligent (DeSI) Research Laboratory, Center for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia. He has authored four monographs, five textbooks, and 300 articles, including 154 refereed international journal articles. His research interests include fuzzy sets and systems, fuzzy optimization, fuzzy transfer learning, and fuzzy modeling in machine learning and data analytics.

Dr. Zhang has won eight Australian Research Council (ARC) Discovery Projects grants and many other research grants. He was awarded the ARC QEII Fellowship. He has co-chaired several international conferences and workshops in the area of fuzzy decision-making and knowledge engineering. He has served as a guest editor for eight special issues of IEEE TRANSACTIONS and other international journals.