Tian Q, Ma C, Zhang FY *et al.* Source-free unsupervised domain adaptation with sample transport learning. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 36(3): 606–616 May 2021. DOI 10.1007/s11390-021-1106-5

Source-Free Unsupervised Domain Adaptation with Sample Transport Learning

Qing Tian[1,2,*], *Member, CCF*, Chuang Ma[1], Feng-Yuan Zhang[1], Shun Peng[1], and Hui Xue[3], *Member, CCF*

[1] *School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China*

[2] *Engineering Research Center of Digital Forensics (Ministry of Education), Nanjing University of Information Science and Technology, Nanjing 210044, China*

[3] *School of Computer Science and Engineering, Southeast University, Nanjing 211189, China*

E-mail: {tianqing, mcboo, zhangfy, pengshun}@nuist.edu.cn; hxue@seu.edu.cn

Received October 22, 2020; accepted April 22, 2021.

Abstract Unsupervised domain adaptation (UDA) has achieved great success in handling cross-domain machine learning applications. It typically benefits the model training of unlabeled target domain by leveraging knowledge from labeled source domain. For this purpose, the minimization of the marginal distribution divergence and conditional distribution divergence between the source and the target domain is widely adopted in existing work. Nevertheless, for the sake of privacy preservation, the source domain is usually not provided with training data but trained predictor (e.g., classifier). This incurs the above studies infeasible because the marginal and conditional distributions of the source domain are incalculable. To this end, this article proposes a source-free UDA which jointly models domain adaptation and sample transport learning, namely Sample Transport Domain Adaptation (STDA). Specifically, STDA constructs the pseudo source domain according to the aggregated decision boundaries of multiple source classifiers made on the target domain. Then, it refines the pseudo source domain by augmenting it through transporting those target samples with high confidence, and consequently generates labels for the target domain. We train the STDA model by performing domain adaptation with sample transport between the above steps in alternating manner, and eventually achieve knowledge adaptation to the target domain and attain confident labels for it. Finally, evaluation results have validated effectiveness and superiority of the proposed method.

Keywords unsupervised domain adaptation, domain shift, sample transport, pseudo source domain

1 Introduction

In machine learning, the models are typically generated on the training data and then deployed on the test data under the hypothesis that the data was sampled from the same statistical distribution, i.e., independent and identical distribution (i.i.d.). However, in many real-world application scenarios, the data used for model training and performance test does not comply with the i.i.d. hypothesis such that the trained model degenerates on the test data. In other words, there exists the divergence between the distributions of the training and the test data, which is distinguished as domain shift [1–3]. To address such distribution shift, the paradigm of unsupervised domain adaptation (UDA) has been introduced, where the data distributions are treated as different domains [4,5]. In UDA, the domain with supervision information (e.g., with labeled data) is distinguished as the source domain but those without supervision knowledge (e.g., with unlabeled data) as the target domain. UDA is aimed to mitigate the domain divergence and leverage knowledge from the source domain to facilitate training the target model [6–8]. One direct way to achieve this goal

Regular Paper

Special Section on Learning from Small Samples

This work was partially supported by the National Natural Science Foundation of China under Grant Nos. 61702273 and 62076062, the Natural Science Foundation of Jinangsu Province of China under Grant No. BK20170956, the Open Projects Program of National Laboratory of Pattern Recognition under Grant No. 20200007, and was also sponsored by Qing Lan Project.

*Corresponding Author

is minimizing the cross-domain conditional distribution inconsistency [9,10]. Along this line, Pan *et al.* [11] proposed the Transfer Component Analysis (TCA) model by transforming both the source and the target domain in dimension-higher Hilbert space, and then performing cross-domain component analysis. Afterwards, Wang *et al.* [12] proposed the BDA method by eliminating the domain shift and tackling the class imbalance.

In some applications, e.g., medical diagnosis [13], the data itself of the source domain is not available because of privacy preservation. In this case, the manner of aligning the domain data distributions is no longer feasible [14–16]. Although the generated source domain models (e.g., the model architectures and parameters) are usually permitted to access, they could not be directly applied to the target domain because of the aforementioned domain shift. To address the problem that the source domain data is unavailable (i.e., the source-free problem), an alternative way is to approximate the source domain from the provided target domain [17,18]. However, two critical issues arise: which target domain data to choose for the source domain approximation, and what strategies to guarantee the approximation confidence.

To address the first issue of approximating the source domain, one often-used strategy is to recover the source domain by predicting the target domain data with the accessible source domain model, and choosing these predicted target domain data far away from the decision boundary as pseudo source points [19,20]. To handle the second issue of characterizing the approximation confidence, the distances of the candidate data points to the decision boundaries have been adopted to measure the approximation confidence [21–23]. Despite both the two issues have been considered in aforementioned methods, the divergence of cross-domain conditional distributions has not been modeled. In addition, they still suffer from the following drawbacks. Firstly, the conditional distributions of their source domains are not consistent with each other such that the desired confident target data is difficult to generate. Secondly, it is difficult to reach consistent confidence decisions between the classifiers of the source domains because of their distribution shift.

To eliminate the drawbacks aforementioned, we propose a novel kind of source-free UDA by taking into account both domain adaptation and sample transport, namely Source Transport Domain Adaptation (STDA). In STDA, one pseudo source domain is firstly constructed according to the aggregated decision boundaries of multiple source classifiers on the target domain. Then, the pseudo source domain is refined by augmenting it through transporting these high-confidence target samples, and in this process the labels of the target domain data are assigned as well. Finally, domain adaptation is achieved with sample transport learning by conducting them in an alternating manner. In summary, the main contributions of this work are four-fold as follows.

1) A source-free UDA method is proposed, coined as Source Transport Domain Adaptation (STDA), which combines domain adaptation and sample transport in the learning process without accessing the source domain data.

2) In STDA, the pseudo source domain is constructed to approximate the source domain according to the boundaries of multiple source domain classifiers on the target domain data.

3) The pseudo source domain is refined through sample transport learning by augmenting it with these confident target domain samples, and in turn assigning labels to these confident target samples in alternating manner.

4) Experimental evaluations are conducted to validate the effectiveness and performance superiority of the proposed method.

The rest of this article is organized as follows. Section 2 introduces preliminary knowledge related to this work. Section 3 elaborates the proposed model and the algorithm. Section 4 reports experimental results and analyzes them. Finally, Section 5 concludes this work and gives future research directions.

2 Preliminaries

2.1 Source-Free UDA

In the past few years, while a variety of UDA methods [5,24] have been proposed, most of them perform domain adaptation from the perspective of distribution alignment between the domains. To estimate the domain distributions, these methods usually require to access the source domain data. Nevertheless, in some real-world application scenarios, the source domain data is not available because of various reasons like privacy protection. This UDA scenario is distinguished as source-free UDA [25,26]. In this case, most of the aforementioned UDA methods cannot work without accessing the source domain data. To address this challenge, model adaptation has been introduced to tackle the source-free UDA [27,28]. Although these methods can achieve UDA, they are significantly different from our proposed STDA method (to be elaborated in Section 3). Specifically, in our STDA method, one target domain associated with multiple trained classifiers is involved, and the decision consistence of these classifiers is aggregated through designing the sample transport rule to guide the choice of the confident target samples for pseudo source domain approximation. In contrast, the aforementioned studies [27,28] approximate the pseudo source domain merely by either one classifier or an adversarial discriminator.

2.2 Domain Distribution Alignment Criteria

In UDA, one of the widely adopted domain alignment measures is the maximum mean discrepancy (MMD) [29–31]. The MMD criterion is mainly dedicated to measure the distribution discrepancy between the domains. Although MMD has been widely incorporated in UDA, it merely measures the marginal distributions of the domains, and cannot characterize the conditional distributions. To handle this issue, the conditional maximum mean discrepancy (CMMD) has been modeled in UDA [2,32]. As aforementioned, both the MMD and the CMMD criteria require accessing the source and the target domain data for estimating the domain distributions. As a result, both the MMD and the CMMD criteria are infeasible in the scenario of source-free UDA.

2.3 Selection and Labeling of Confident Target Instances

In source-free UDA, one way of recovering the source domain is approximating it with the confident target domain samples [33,34]. For this purpose, the label confidence model was proposed by labeling the target instances with soft confidence, and then applying them as the source domain surrogate to measure cross-domain distribution divergence [33]. Another representative way is the selection bias mechanism [34]. Specifically, it measures the domain divergence by estimating the bias of the selected target instances. Recently, active learning (AL) [35,36] has also been employed in domain adaptation by labeling the target instances by the Oracle. Although the AL technique can effectively supervise the domain adaptation by labeling informative data instances, it was usually adopted in semi-supervised rather than unsupervised domain adaptation.

3 Proposed Method

In this section we propose an unsupervised source-free domain adaptation method, called Source Transport Domain Adaptation (STDA), which performs inductive domain knowledge transfer from the provided source classifier rather than the source training data.

3.1 Problem Setting

In unsupervised domain adaptation, the source domain is represented as $\mathcal{D}_s = \{\boldsymbol{X}_s, \boldsymbol{Y}_s\} = \{(\boldsymbol{x}_i^s, \boldsymbol{y}_i^s)\}_{i=1}^M$ while the target domain is represented as $\mathcal{D}_t = \{\boldsymbol{X}_t\} = \{\boldsymbol{x}_j^t\}_{j=1}^N$, where $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ are the input space and $\boldsymbol{Y}_s = \{1, 2, \cdots, C\}$ is the label space with M (N) and C being the number of the corresponding domain samples and classes, respectively. In addition, let $p(\boldsymbol{X}_s)$ and $q(\boldsymbol{X}_t)$ denote the marginal distributions of the source and the target domains, respectively. In the setting of source-free UDA, $p(\boldsymbol{X}_s) \neq q(\boldsymbol{X}_t)$. It should be noted that $p(\boldsymbol{X}_s)$ cannot be directly estimated since the source domain data is unavailable. Instead, we are given K trained source domain classifiers, characterized by $\{\theta_k\}_{k=1}^K$.

3.2 Source-Free Domain Adaptation

Domain adaptation aims to leverage source domain knowledge to benefit the training of the target domain tasks. However, since the source and the target domains typically do not comply with the same i.i.d. distributions, domain alignment is usually required to alleviate their distribution divergence. One of the widely-used strategies is to align the domains by minimizing their maximum mean discrepancy (MMD):

$$MMD^2\left(\mathcal{D}_s, \mathcal{D}_t\right) = \left\| \frac{1}{M}\sum_{i=1}^{M}\phi\left(\boldsymbol{x}_i^s\right) - \frac{1}{N}\sum_{j=1}^{N}\phi\left(\boldsymbol{x}_j^t\right) \right\|_{\mathcal{H}}^2, \quad (1)$$

where $\phi(\cdot)$ denotes the feature mapping (e.g., the Gaussian kernel function) in the RKHS space $\mathcal{H}$. Unfortunately, when the source domain data is not available or not accessible (i.e., source-free), (1) cannot be estimated directly. To address this challenge, we assume that all the target domain instances are pretended as the source domain whose labels are assigned by the source domain classifiers $\{\theta_k\}_{k=1}^K$. In this setting, each of the target domain instances will be predicted by the K source classifiers (taking the i-th instance $\boldsymbol{x}_i^t$ as example): $\{\theta_1(\boldsymbol{x}_i^t), \theta_2(\boldsymbol{x}_i^t), \cdots, \theta_K(\boldsymbol{x}_i^t)\}$. According to the decision confidence of the target domain instances by the source domain classifiers, we can approximate the source domain with its pseudo $p\mathcal{D}_s$. Then, (1) can be reformulated as

$$\begin{aligned} & MMD^2\left(p\mathcal{D}_s, \mathcal{D}_t\right) \\ = & \left\| \frac{1}{M}\sum_{i=1}^{M}\phi\left(\boldsymbol{x}_i^{ps}\right) - \frac{1}{N}\sum_{j=1}^{N}\phi\left(\boldsymbol{x}_j^{t}\right) \right\|_{\mathcal{H}}^2 . \end{aligned} \tag{2}$$

It is obvious that there is no distribution discrepancy in (2) if the target domain completely equals the source domain. Moreover, without consideration of the instance labels, the distribution of the built pseudo source domain is always the same as that of the target domain, which makes (2) meaningless in addressing the source-free UDA. However, when the conditional distribution is considered, MMD will be extended to the conditional MMD (CMMD), shown as follows:

$$\begin{aligned} & CMMD^2\left(\mathcal{D}_s, \mathcal{D}_t\right) \\ = & \sum_{c=1}^{C}\left\| \frac{1}{M_c}\sum_{i=1}^{M_c}\phi\left(\boldsymbol{x}_i^{s_c}\right) - \frac{1}{N_c}\sum_{j=1}^{N_c}\phi\left(\boldsymbol{x}_j^{t_c}\right) \right\|_{\mathcal{H}}^2 , \end{aligned} \tag{3}$$

where M_c and N_c represent the number of source instances and target instances in the c-th class, respectively. $\boldsymbol{x}_i^{s_c}$ and $\boldsymbol{x}_j^{t_c}$ denote the i-th and the j-th instances from the c-th class in the source and the target domains, respectively. In fact, if we calculate the class centers of the source domain by (3), there will be tiny errors due to the uncertainty of the source sample labels. If all the samples have the only one-hot label, (3) is a true representation. By contrast, when the samples are encoded with soft labels, the center of source class c will be represented as follows:

$$center(c) = \frac{1}{M}\sum_{i=1}^{M}\theta^c(\phi\left(\boldsymbol{x}_i^s\right))\phi\left(\boldsymbol{x}_i^s\right), \tag{4}$$

where $\theta^c(\phi\left(\boldsymbol{x}_i^s\right))$ means the probability of the sample belonging to the corresponding class. In (4), the center of each class is modified by each sample. In soft labels, all the samples will affect each class center. As a result, the source domain class center will change after the source domain data updating, leading to new distribution divergence between the domains. To address this issue, we assume all the classifiers apply the soft labels or the class probability $\theta_i(\boldsymbol{x}) \in \mathbb{R}^C$, and then the class probability will become the decision criterion. With the class probability, we can reconstruct the pseudo source domain $p\mathcal{D}_s = \{\boldsymbol{x}_i^t, \theta_{\max}(\boldsymbol{x}_i^t)\}_{i=1}^N$, where $\theta_{\max}(\boldsymbol{x})$ indicates the confidence-highest classifier. Through combining the pseudo source domain with (3), we have the following equation:

$$\begin{aligned} & CMMD^2\left(p\mathcal{D}_s, \mathcal{D}_t\right) \\ = & \sum_{c=1}^{C}\left\| \frac{1}{N}\sum_{i=1}^{N}\theta_{\max}^c(\boldsymbol{x}_i^{ps})\phi\left(\boldsymbol{x}_i^{ps}\right) - \frac{1}{N_c}\sum_{j=1}^{N_c}\phi\left(\boldsymbol{x}_j^{t_c}\right) \right\|_{\mathcal{H}}^2 , \end{aligned} \tag{5}$$

where the labels of the target instances $\boldsymbol{x}_j^{t_c}$ are assigned by the confidence-highest classifier. $\theta_{\max}^c(\boldsymbol{x}_i^{ps})$ indicates the probability of the i-th sample belonging to the c class on the most reliable classifiers on the source domain. It is worth noting that the labels on target samples are the one-hot labels while the labels on pseudo source samples are soft labels. By further taking advantages of matrix tricks and regularization, (5) can be extended as follows:

$$\begin{aligned} & \min_{\boldsymbol{A}} \operatorname{tr}\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}\ \sum_{c=1}^{C}\boldsymbol{M}_c\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\right) + \lambda\|\boldsymbol{A}\|_F^2 \\ & \text{s.t. } \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{H}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{I}, \end{aligned} \tag{6}$$

where

$$(M_c)_{ij} = \begin{cases} \dfrac{\theta_{\max}^c(\boldsymbol{x}_i^{ps})}{N^2}, & \text{if } \boldsymbol{x}_i, \boldsymbol{x}_j \in p\mathcal{D}_s, \\ \dfrac{1}{N_c^2}, & \text{if } \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{D}_t^{(c)}, \\ -\dfrac{\theta_{\max}^c(\boldsymbol{x}_i^{ps})}{NN_c}, & \text{if } \begin{cases} \boldsymbol{x}_i \in p\mathcal{D}_s, \boldsymbol{x}_j \in \mathcal{D}_t^{(c)}, \\ \boldsymbol{x}_i \in \mathcal{D}_t^{(c)}, \boldsymbol{x}_j \in p\mathcal{D}_s, \end{cases} \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

$\boldsymbol{A}$ denotes the transformation matrix, and $\boldsymbol{I}$ is the $2N$-order identity matrix. In addition, $\boldsymbol{H} = \boldsymbol{I} - \frac{1}{2N}\mathbf{1}$ is the centering matrix with $\mathbf{1}$ being an all-one matrix. By minimizing (6) with (7), we could achieve a proper pseudo source domain, which can better approximate the marginal distribution of the source domain.

3.3 Sample Selection and Transport

In regular unsupervised domain adaptation, the labeled data is only available in the source domain. And in Subsection 3.2, a method adjusting the target domain has been proposed. Moreover, the regular selection strategy is to find the instance which has the same label in different classifiers. However, there is no evidence that the target domain after tuning-up is the closest domain to the unavailable source domain and the label is correct enough. To this end, a strategy to measure the accuracy of sample classification is proposed. In sample selection, we set up a sample transport rule with the help of sample transport learning.

When designing the rule, we want to implement the following two functions: firstly, the label probability on each classifier should be accurate enough; secondly, the classification gap of each classifier should be as small as possible. Based on these considerations, we design the following sample transport rule:

$$ST(\boldsymbol{x}) = \frac{1}{K}\frac{\sum_{k=1}^{K}\max(\theta_k(\boldsymbol{x}))}{\frac{1}{2}\sum_{i,j=1}^{K}||\theta_i(\boldsymbol{x})-\theta_j(\boldsymbol{x})||_2^2}. \tag{8}$$

From (8), we can find that the above two considerations are both addressed by measuring $ST(\boldsymbol{x})$. Generally, a proper threshold setting will help achieve a better result.

3.4 Sample Transport Domain Adaptation

By taking into account the above considerations (in Subsections 3.2 and 3.3), the proposed STDA can be naturally constructed, with two learning steps as follows.

1) Based on (6), we have a quantitative method to measure the matching degree between the target domain and the source domain classifiers. Then the Lagrange multiplier method will be used to solve (6), which is shown as follows:

$$\left(\boldsymbol{X}\sum_{c=1}^{C}\boldsymbol{M}_c\boldsymbol{X}^{\mathrm{T}}+\lambda\boldsymbol{I}\right)\boldsymbol{A}=\boldsymbol{X}\boldsymbol{H}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\Phi, \tag{9}$$

where Φ is the Lagrange multiplier. Then the transformation matrix $\boldsymbol{A}$ can be obtained as the eigenvectors corresponding to the smallest d eigenvalues of (9).

2) After obtaining the transformation matrix $\boldsymbol{A}$, all the target domain instances will be transformed as $\{\boldsymbol{A}\boldsymbol{x}_i^t\}_{i=1}^N$. At this point, all target instances are subject to supervision by sample transport rule. If $ST(\boldsymbol{A}\boldsymbol{x}_i^t)\geqslant b$ where b is the transport threshold, the instance $\boldsymbol{A}\boldsymbol{x}_i^t$ will be treated as a confident sample of which the label will be replaced by one-hot label instead of soft label and not to be updated anymore. Then the sample $\boldsymbol{x}_i^{ps}$ will be replaced by $\boldsymbol{A}\boldsymbol{x}_i^t$. This step will be iterated until all the samples have been tested by the transport rule.

In summary, the complete process of the STDA algorithm is elaborated in Algorithm 1, where $\boldsymbol{x}_{i,\,iter}^t$ means the instance in the $iter$-th iteration and $\boldsymbol{x}_{i,0}^t$ is the initialization. We can see that the STDA algorithm is an inductive algorithm, since it can generalize to unseen test data with the output transformation matrix $\boldsymbol{A}$. In addition, for the sake of better understanding, the algorithm process is also demonstrated in Fig.1.

Algorithm 1. Algorithm for STDA

Input: $\{\boldsymbol{x}_{i,0}^t\}_{i=1}^N$: training data for the target domain;
$\{\theta_i\}_{i=1}^K$: different classifiers on the source domain;
b: the threshold in the sample transport rule;
$iter_{\max}$: the maximum number of iterations
Output: the transformation matrix $\boldsymbol{A}$;
Initialize the pseudo source domain by classifiers $p\mathcal{D}_s=\{\boldsymbol{x}_i^{ps},\boldsymbol{y}_i^{ps}\}_{i=1}^N=\{\boldsymbol{x}_{i,0}^t,\theta_{\max}(\boldsymbol{x}_{i,0}^t)\}_{i=1}^N$ where the pseudo source soft label $\boldsymbol{y}_i^{ps}=\{\theta_{\max}(\boldsymbol{x}_{i,0}^t)\}_{i=1}^N$; and the iteration $iter=0$;
while $iter < iter_{\max}$ **do**
  Update the transformation $\boldsymbol{A}$ by (9);
  Update $\{\boldsymbol{x}_{i,iter+1}^t\}_{i=1}^N=\{\boldsymbol{A}\boldsymbol{x}_{i,iter}^t\}_{i=1}^N$;
  foreach target sample $\boldsymbol{x}_{i,iter+1}^t$ **do**
    if $\boldsymbol{y}_i^{ps}$ is a soft label and $ST(\boldsymbol{x}_{i,iter+1}^t)\geqslant b$ by (8) **then**
      $p\mathcal{D}_s=p\mathcal{D}_s\backslash\boldsymbol{x}_i^{ps}$;
      $p\mathcal{D}_s=p\mathcal{D}_s\cup\boldsymbol{x}_{i,iter+1}^t$;
      Assign $\boldsymbol{y}_i^{ps}$ with one-hot label whose entity with maximal probability is 1;
    end
  end
  $iter=iter+1$.
end

It is worth noting that with the manner in (5) trained via Algorithm 1, the source and the target domain can be aligned closer to each other through minimizing the divergence between the generated pseudo source domain and the target domain. More importantly, their conditional divergence can be effectively measured as well. The explanations are as follows. For the target domain, all its samples are transformed by $\boldsymbol{A}$ in each iteration of Algorithm 1. By comparison, for the source domain only these pseudo source samples that pass through the transport rule in current iteration will be updated by $\boldsymbol{A}$. In addition, the generated pseudo source samples are encoded with soft labels while the target samples are with one-hot labels. As a result, the approximated source domain is substantially not identical to the target domain, which makes (5) reasonable.

4 Experiments

In this section, we conduct experiments to evaluate the effectiveness of the proposed STDA method.

4.1 Datasets

The experiments are conducted on the Office-10+Caltech-10 dataset. For Office-10+Caltech-10, it has been widely used in DA. The Office-31 and Caltech-

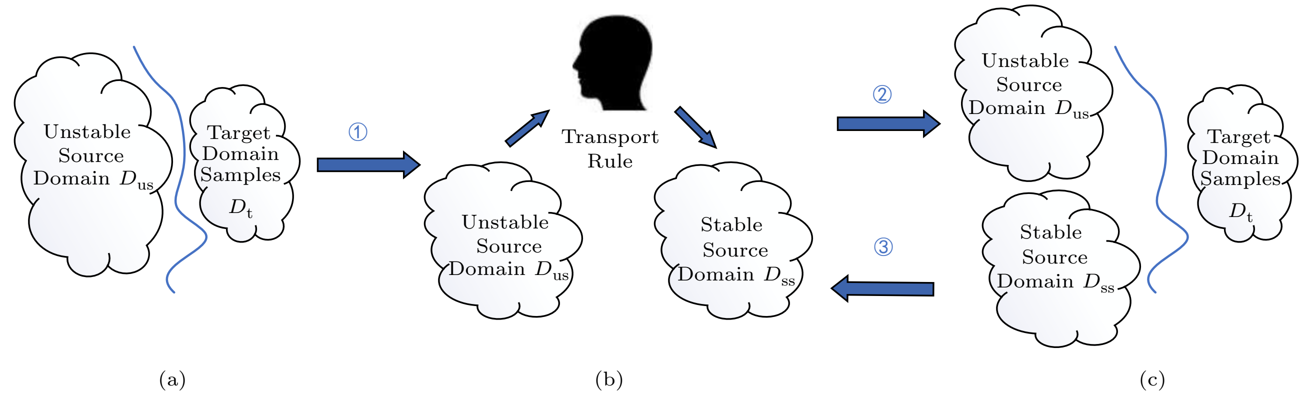


Fig.1. Work-flow of the STDA algorithm. (a) The source domain is initialled with pseudo samples and unstable, and (b) with increased iterations of sample transport learning, some of the unstable samples are transformed into stable ones, and (c) the domain adaptation is executed between the updated pseudo source domain and the target domain until the above two steps converge.

256 formed by the same 10 classes are extracted from Office-31 and Caltech-256, respectively. It is generally believed that the Office-10 dataset contains four domains: Amazon (A), Webcam (W) and DSLR (D) from Office-10, and Caltech (C) from Caltech-10. Besides, additional experiments on the digit handwritten datasets, i.e. the USPS (U) dataset and the MNIST (M) dataset are also performed. The USPS and MNIST datasets are standard digital recognition datasets containing handwriting digits from 0 to 9. The details of each dataset are shown in Table 1.

4.2 Comparison Methods and Setup

4.2.1 Comparison Methods

Some methods are introduced to compare with STDA.

• *1-NN*. 1-NN is a basic classification algorithm, that is, given a training dataset, for a new input instance, we find the instance closest to the instance in the training dataset. This instance belongs to a certain class, and then the input instance will be classified into this class.

• *SVM*. SVM is also a basic classification algorithm which is to solve the separation hyperplane which can divide the training data correctly.

• *Geodesic Flow Kernel (GFK)* [37]. GFK is a domain adaptation method which constructs a geodesic to make the source domain close to the target domain in Grassmann manifold.

• *Transfer Component Analysis (TCA)* [11]. TCA tries to find a mapping, which can project both the source domain and the target domain into a high latitude Hilbert space, and then calculates the mean distance between the two stacks of data after projection.

• *Joint Distribution Adaptation (JDA)* [38]. JDA is an evolution of TCA which combines TCA and conditional distribution and reduces the distribution difference between domains.

• *Balance Distribution Adaptation (BDA)* [12]. BDA leverages the importance of the marginal and conditional distribution discrepancies and balances different domains.

• *Easy Transfer Learning (EasyTL)* [39]. EasyTL is the easy method which requires no model selection and hyperparameter tuning, while achieving competitive performance.

4.2.2 Experimental Setup

In the experiments, the domains are represented with SURF features. Initialization of the target pseudo labels is achieved with the maximum classification probabilities of the classifiers, and then updated by previous round results. In addition, we adopt SVM, the Bayes classifier and the neural network classifier as

Table 1. Details of Experimental Datasets

Dataset	Type	Number of Classes	Feature Dimension	Number of Samples	Domain
Office-10	Object	10	800	1 410	A, W, D
Caltech-10	Object	10	800	1 123	C
USPS	Digit	10	256	1 800	U
MNIST	Digit	10	256	2 000	W

three classifiers, and encode their output probabilities as soft labels. When reaching the maximum iteration, instances failing to pass through the transport rule will achieve the most reliable label on all classifiers. For feasible comparison, the data class centers of the source domain are provided for the compared methods (except for our method), and the parameters in each compared method are assigned according to the settings of corresponding reference. For our STDA method, both the threshold b and the parameter λ are set to 1 by prior knowledge. All the reported results are averaged over 10 random runs[1].

4.3 Results and Analysis

4.3.1 *Results on Office-10 + Caltech-10*

We test the performance of our method STDA and the other methods on Office-10 + Caltech-10. The results are reported in Table 2. In Table 2, we have the following findings.

• In 10 of the 12 cases, STDA achieves the best results. Specially, in W $\rightarrow$ D and D $\rightarrow$ W tasks, where "$\rightarrow$" means the knowledge transfer from the source domain to the target domain, STDA gets accuracy 95.83% and 94.24% which is much higher than those of other methods due to the fact that the Webcam dataset and the Dslr dataset have the similar feature maps, and the classifiers of one dataset will perform well on the others.

• Compared with the single-classifier method, STDA performs the best. This is because sample transport will screen the reliable samples and change the distribution of pseudo source domain. It has been proved that the prediction considering the structural risk is better than the single prediction.

• Compared with the other domain adaptation methods, due to the lack of constraints on other classification boundaries, these methods, i.e., TCA, JDA, BDA and EasyTL, do not perform well. This implies that multiple independent decisions will improve the classification results.

Also, other results on handwritten datasets are shown in Fig.2. The results show that our method has no significant advantages, which may result from the fact that the similarity of classification boundaries leads to the same classification results.

4.3.2 *Analysis on Convergence and Accuracy*

Compared with the other performance measures, the convergence efficiency is an importance index due to the intuitive representation of run time. The accuracy with iterations of different methods is shown in Fig.3. It is noting that EasyTL has no iterations, which has the same property with TCA, so we only show the iteration of TCA without EasyTL. We observe the following findings.

• From the convergence, STDA converges gradually to a certain extent. The evidence is that the accuracy remains virtually unchanged after several iterations.

• The accuracy of TCA remains unchanged with iterations, since it is solved in one iteration with closed-form solution. In addition, the accuracies of JDA and BDA ascend in the first iteration but then dither severely with iterations. It is because their solution paths are not stable in the optimization process. By contrast, with increased iterations, the accuracy of STDA steadily ascends. It validates the effectiveness and superiority of this algorithm in gradually augmenting the source domain using the confident target data through sample transport.

Table 2. Estimation Accuracy (%) of STDA and Other Methods on Office-10 + Caltech-10 Dataset

Method	C $\rightarrow$ A	C $\rightarrow$ W	C $\rightarrow$ D	A $\rightarrow$ C	A $\rightarrow$ W	A $\rightarrow$ D	W $\rightarrow$ C	W $\rightarrow$ A	W $\rightarrow$ D	D $\rightarrow$ C	D $\rightarrow$ A	D $\rightarrow$ W
1-NN	23.70	25.76	25.48	26.00	29.83	25.48	19.86	22.96	59.24	26.27	28.50	63.39
SVM	36.95	32.54	38.22	34.73	35.59	27.39	26.36	31.00	70.07	29.65	32.05	75.93
GFK	39.53	39.42	34.64	37.80	36.21	34.57	23.68	26.94	77.36	28.47	30.44	74.23
TCA	40.35	33.79	43.11	38.53	36.46	30.04	23.29	28.73	81.46	30.08	29.60	82.57
JDA	42.41	35.24	40.54	35.27	32.66	38.58	21.57	29.64	83.02	30.56	31.83	75.80
BDA	42.43	35.21	43.25	37.57	35.03	39.80	24.63	30.64	83.02	30.50	**31.25**	76.32
EasyTL	43.64	36.51	43.59	38.06	35.86	39.80	22.64	**31.95**	83.02	31.45	30.46	80.31
STDA	**48.83**	**39.53**	**46.26**	**41.53**	**41.39**	**44.87**	**29.75**	30.72	**95.83**	**37.21**	30.52	**94.24**

Note: Best results are in bold.

[1]The source codes of STDA and the compared methods can be found at https://github.com/mc-boo/sample-transport-domain-adaptation and https://github.com/jindongwang/transferlearning respectively.

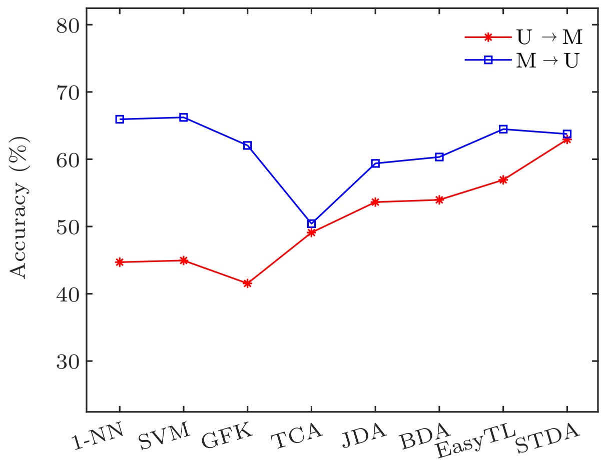


Fig.2. Estimation accuracy (%) on digit handwriting datasets.

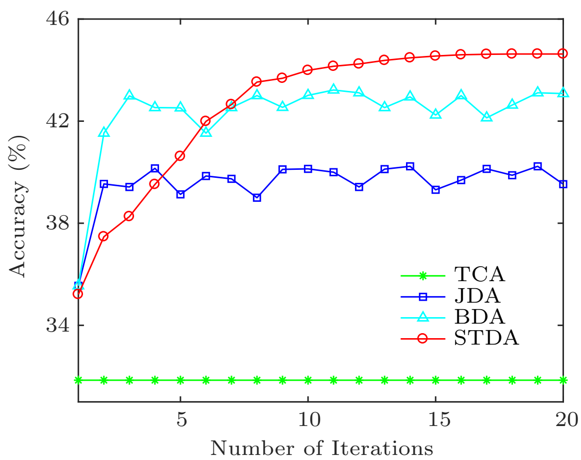


Fig.3. Accuracy comparison with increased iterations on A → D.

• From the perspective of change rate, STDA has a smaller rate in higher iteration. This is intuitive and can be explained. The sample transport rule will identify samples in each round, and the remaining samples are more and more difficult to identify after many iterations, which leads to the gradual decrease of correct samples in each round, and finally leads to no increase in the accuracy rate.

4.3.3 Analysis on Different Classifiers

In this subsection, comparison experiments are performed to confirm the accuracy of the proposed method. As have mentioned in previous experiments, three classifiers are used to achieve the best performance. This subsection demonstrates the combination of classifiers. The results are shown in Table 3, where we can find that 1-NN achieves the worst result, and its hybrid algorithm cannot achieve the best performance. When considering the function of soft label, we can infer that the hard label will not be suitable for our algorithm and that is why we choose the soft label for the confidence. Moreover, in some domain adaptation tasks, the results of multiple classifiers are worse than those of single classifier, maybe because too many classifier boundaries confuse the classification subspace, which leads to inaccurate class of samples. Therefore, it is very important to select the appropriate classifier and the classification boundary.

4.3.4 Ablation Study

In this subsection, an ablation study is performed to evaluate the components of STDA. The ablation experiment is conducted from two aspects: 1) without conditional MMD (w/o CMMD) and 2) without sample transport rule (w/o $ST(\boldsymbol{x})$). For the conditional MMD, if we remove it from the STDA objective (i.e., the first term in (6)), the eventual results will be determined by the adopted three classifiers without domain alignment. It means the STDA model degenerates to an ordinary ensemble learning method. For the sample transport rule (see (8)), if we remove it from the STDA algorithm (see Algorithm 1), the labels of the target instances will be determined by the classifier with the maximal probability. The ablation experiment is preformed on the Office-10+Caltech-10 dataset with results reported in Table 4.

Table 3. Estimation Accuracy (%) of STDA with Different Classifiers

Method	C → A	C → W	C → D	A → C	A → W	A → D	W → C	W → A	W → D	D → C	D → A	D → W
1-NN	23.70	25.76	25.48	26.00	29.83	25.48	19.86	22.96	59.24	26.27	28.50	63.39
SVM	36.95	32.54	38.22	34.73	35.59	27.39	26.36	31.00	70.07	29.65	32.05	75.93
NN	37.32	34.62	36.91	35.10	37.55	26.48	25.75	31.96	71.38	30.00	31.52	78.94
Bayes	34.65	30.52	28.94	30.08	30.74	27.90	24.64	28.65	68.53	28.97	30.19	74.23
1-NN+SVM+NN	47.52	38.56	44.23	39.75	37.08	41.60	25.64	27.56	90.43	36.74	27.67	90.04
1-NN+NN+Bayes	46.23	37.69	43.60	38.62	36.64	40.61	29.67	29.75	89.58	36.87	26.85	89.50
1-NN+SVM+Bayes	47.59	37.51	45.23	39.65	35.99	39.06	27.50	26.42	90.12	35.77	28.34	91.43
SVM+NN+Bayes	**48.83**	**39.53**	**46.26**	**41.53**	**41.39**	**44.87**	**29.75**	**30.72**	**95.83**	**37.21**	**30.52**	**94.24**

Note: Best results are in bold.

Table 4. Ablation Study on Office-10 + Caltech-10 Dataset

Method	C → A	C → W	C → D	A → C	A → W	A → D	W → C	W → A	W → D	D → C	D → A	D → W	Mean
w/o CMMD	43.42	38.64	37.57	35.61	40.25	31.84	27.87	**31.52**	57.32	26.89	24.21	56.30	37.62
w/o $ST(\boldsymbol{x})$	42.73	34.14	43.28	37.24	39.32	36.39	25.74	24.65	70.35	34.26	27.45	73.25	40.73
STDA	**48.83**	**39.53**	**46.26**	**41.53**	**41.39**	**44.87**	**29.75**	30.72	**95.83**	**37.21**	**30.52**	**94.24**	**48.39**

Note: Best results are in bold.

We can observe the following findings. Without the CMMD component, the STDA method (w/o CMMD) achieves the lowest accuracy (37.62% mean accuracy). Comparatively, when removing the transport rule, the STDA method (w/o $ST(\boldsymbol{x})$) reaches better results (40.73% mean accuracy). Interestingly, when both the above components are incorporated in the STDA algorithm, significant performance improvements are yielded (48.39% mean accuracy). It means that both the CMMD domain alignment and the sample transport learning effectively bring about performance benefits, which validates the rationality of our method.

4.3.5 Parameter Sensitivity of the Transport Threshold

As demonstrated in Algorithm 1, the sample transport rule is critical in approximating the source domain. To evaluate it, we perform sensitivity experiment on its hyperparameter b. Without loss of generality, we demonstrate some results on the task C → A in Fig.4.

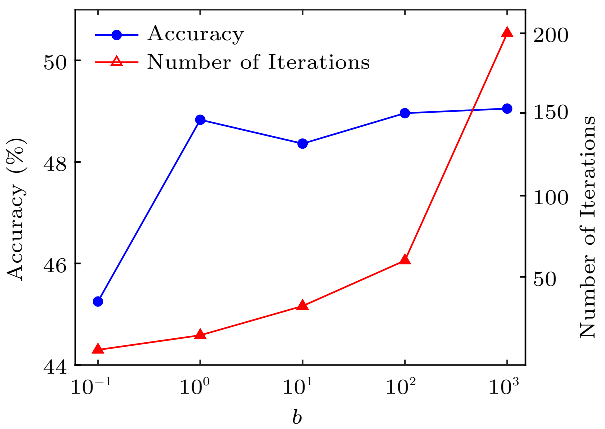


Fig.4. Accuracy and convergence iterations of the STDA algorithm with increased threshold b.

We can observe that with increased b, the number of iterations required for algorithm convergence grows exponentially, while the accuracy stays nearly unchanged when $b \geqslant 1$. The reason is that a larger b incurs much fewer samples passing through the transport rule in each iteration. That is the reason why we set $b = 1$ in the experimental setup.

5 Conclusions

This article proposed a novel kind of source-free unsupervised domain adaptation method, named Sample Transport Domain Adaptation (STDA), by modeling domain adaptation with sample transport. The STDA model works well without accessing the source domain data, by which the source data privacy can be preserved. Specifically, STDA firstly estimates the pseudo source domain and its pseudo labels. Secondly, it seeks the most valuable instances for labeling through an effective selection strategy. Thirdly, it updates the cross-domain distribution alignment between the updated pseudo source and the target domain. After several alternating rounds between above steps, consequently reliable labels can be generated for the target domain. Experimental evaluations demonstrate the effectiveness and superiority of the proposed STDA method. However, the proposed model is shallow and may be not able to completely explore those highly nonlinear knowledge from the domains. Therefore, in the future, we will consider to extend the proposed model with deep neural networks.

References

[1] Pan S J, Yang Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2009, 22(10): 1345-1359. DOI: 10.1109/TKDE.2009.191.

[2] Yan H, Ding Y, Li P, Wang Q, Xu Y, Zuo W. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *Proc. the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, July 2017, pp.2272-2281. DOI: 10.1109/CVPR.2017.107.

[3] Tahmoresnezhad J, Hashemi S. Visual domain adaptation via transfer feature learning. *Knowledge and Information Systems*, 2017, 50(2): 585-605. DOI: 10.1007/s10115-016-0944-x.

[4] Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, Marchand M, Lempitsky V. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 2016, 17(1): 2096-2030. DOI: 10.1007/978-3-319-58347-1_10.

[5] Ganin Y, Lempitsky V. Unsupervised domain adaptation by backpropagation. In *Proc. the 32nd International Conference on Machine Learning*, July 2015, pp.1180-1189.

[6] Saito K, Watanabe K, Ushiku Y, Harada T. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proc. the 2018 IEEE Conference on Computer Vision and Pattern Recognition*, June 2018, pp.3723-3732. DOI: 10.1109/CVPR.2018.00392.

[7] Baktashmotlagh M, Harandi M T, Lovell B C, Salzmann M. Unsupervised domain adaptation by domain invariant projection. In *Proc. the 2013 IEEE International Conference on Computer Vision*, December 2013, pp.769-776. DOI: 10.1109/ICCV.2013.100.

[8] Pan Y, Yao T, Li Y, Wang Y, Ngo C W, Mei T. Transferrable prototypical networks for unsupervised domain adaptation. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019, pp.2239-2247. DOI: 10.1109/CVPR.2019.00234.

[9] Lee C Y, Batra T, Baig M H, Ulbricht D. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019, pp.10285-10295. DOI: 10.1109/CVPR.2019.01053.

[10] Lee S, Kim D, Kim N, Jeong S G. Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In *Proc. the 2019 IEEE/CVF International Conference on Computer Vision*, Oct. 27–Nov. 2, 2019, pp.91-100. DOI: 10.1109/ICCV.2019.00018.

[11] Pan S J, Tsang I W, Kwok J T, Yang Q. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 2010, 22(2): 199-210. DOI: 10.1109/TNN.2010.2091281.

[12] Wang J, Chen Y, Hao S, Feng W, Shen Z. Balanced distribution adaptation for transfer learning. In *Proc. the 2017 IEEE International Conference on Data Mining*, November 2017, pp.1129-1134. DOI: 10.1109/ICDM.2017.150.

[13] Kononenko I. Machine learning for medical diagnosis: History, state of the art and perspective. *Artificial Intelligence in Medicine*, 2001, 23(1): 89-109. DOI: 10.1016/S0933-3657(01)00077-X.

[14] Chen F, Bruhadeshwar B, Liu A X. Cross-domain privacy-preserving cooperative firewall optimization. *IEEE/ACM Transactions on Networking*, 2012, 21(3): 857-868. DOI: 10.1109/TNET.2012.2217985.

[15] Lee T, Pappas C, Barrera D, Szalachowski P, Perrig A. Source accountability with domain-brokered privacy. In *Proc. the 12th International Conference on Emerging Networking Experiments and Technologies*, December 2016, pp.345-358. DOI: 10.1145/2999572.2999581.

[16] Zhang L, Zhang D. Domain adaptation extreme learning machines for drift compensation in E-nose systems. *IEEE Transactions on Instrumentation and Measurement*, 2014, 64(7): 1790-1801. DOI: 10.1109/TIM.2014.2367775.

[17] Kim Y, Hong S, Cho D, Park H, Panda P. Domain adaptation without source data. arXiv:2007.01524, 2020. https://arxiv.org/abs/2007.01524v2, January 2021.

[18] Long M, Cao Z, Wang J, Jordan M I. Conditional adversarial domain adaptation. In *Proc. the 32nd International Conference on Neural Information Processing Systems*, December 2018, pp.1640-1650.

[19] Hu J, Mo Q, Liu Z *et al.* Multi-source classification: A DOA-based deep learning approach. In *Proc. the 2020 International Conference on Computer Engineering and Application*, March 2020, pp.463-467. DOI: 10.1109/ICCEA50009.2020.00106.

[20] Zhao C, Wang S, Li D. Multi-source domain adaptation with joint learning for cross-domain sentiment classification. *Knowledge-Based Systems*, 2020, 191: Article No. 105254. DOI: 10.1016/j.knosys.2019.105254.

[21] Wang J, Feng W, Chen Y, Yu H, Huang M, Yu P S. Visual domain adaptation with manifold embedded distribution alignment. In *Proc. the 26th ACM International Conference on Multimedia*, October 2018, pp.402-410. DOI: 10.1145/3240508.3240512.

[22] Wang J, Chen Y, Feng W, Yu H, Huang M, Yang Q. Transfer learning with dynamic distribution adaptation. *ACM Transactions on Intelligent Systems and Technology*, 2020, 11(1): Article No. 6. DOI: 10.1145/3360309.

[23] Behrend R E, Pearce P A, Petkova V B, Zuber J B. On the classification of bulk and boundary conformal field theories. *Physics Letters B*, 1998, 444(1/2): 163-166. DOI: 10.1016/S0370-2693(98)01374-4.

[24] Wang Q, Breckon T. Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In *Proc. the 34th AAAI Conference on Artificial Intelligence*, February 2020, pp.6243-6250. DOI: 10.1609/aaai.v34i04.6091.

[25] Kundu J N, Venkat N, Babu R V *et al.* Universal source-free domain adaptation. In *Proc. the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2020, pp.4544-4553. DOI: 10.1109/CVPR42600.2020.00460.

[26] Nelakurthi A R, Maciejewski R, He J. Source free domain adaptation using an off-the-shelf classifier. In *Proc. the 2018 IEEE International Conference on Big Data*, December 2018, pp.140-145. DOI: 10.1109/BigData.2018.8622112.

[27] Duan L, Tsang I W, Xu D, Chua T S. Domain adaptation from multiple sources via auxiliary classifiers. In *Proc. the 26th International Conference on Machine Learning*, June 2009, pp.289-296. DOI: 10.1145/1553374.1553411.

[28] Zhang Y, Tang H, Jia K, Tan M. Domain-symmetric networks for adversarial domain adaptation. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019, pp.5031-5040. DOI: 10.1109/CVPR.2019.00517.

[29] Dziugaite G K, Roy D M, Ghahramani Z. Training generative neural networks via maximum mean discrepancy optimization. arXiv:1505.03906, 2015. https://arxiv.org/abs/1505.03906, January 2021.

[30] Iyer A, Nath S, Sarawagi S. Maximum mean discrepancy for class ratio estimation: Convergence bounds and kernel selection. In *Proc. the 31st International Conference on Machine Learning*, June 2014, pp.530-538.

[31] Li J, Zhao J, Lu K. Joint feature selection and structure preservation for domain adaptation. In *Proc. the 25th International Joint Conference on Artificial Intelligence*, July 2016, pp.1697-1703.

[32] Chen Y, Song S, Li S, Wu C. A graph embedding framework for maximum mean discrepancy-based domain adaptation algorithms. *IEEE Transactions on Image Processing*, 2019, 29: 199-213. DOI: 10.1109/TIP.2019.2928630.

[33] Donmez P, Carbonell J G, Schneider J. Efficiently learning the accuracy of labeling sources for selective sampling. In *Proc. the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, June 2009, pp.259-268. DOI: 10.1145/1557019.1557053.

[34] Liu A, Ziebart B. Robust classification under sample selection bias. In *Proc. the 27th International Conference on Neural Information Processing Systems*, December 2014, pp.37-45.

[35] Huang S J, Li G X, Huang W Y, Li S Y. Incremental multi-label learning with active queries. *Journal of Computer Science and Technology*, 2020, 35(2): 234-246. DOI: 10.1007/s11390-020-9994-3.

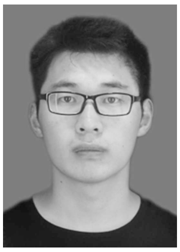
[36] Gao N, Huang S J, Yan Y, Chen S. Cross modal similarity learning with active queries. *Pattern Recognition*, 2018, 75: 214-222. DOI: 10.1016/j.patcog.2017.05.011.

[37] Gong B, Shi Y, Sha F, Grauman K. Geodesic flow kernel for unsupervised domain adaptation. In *Proc. the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp.2066-2073. DOI: 10.1109/CVPR.2012.6247911.

[38] Long M, Wang J, Ding G, Sun J, Yu P S. Transfer feature learning with joint distribution adaptation. In *Proc. the 2013 IEEE International Conference on Computer Vision*, December 2013, pp.2200-2207. DOI: 10.1109/ICCV.2013.274.

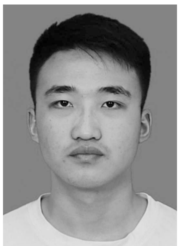
[39] Wang J, Chen Y, Yu H, Huang M, Yang Q. Easy transfer learning by exploiting intra-domain structures. In *Proc. the 2019 IEEE International Conference on Multimedia and Expo*, July 2019, pp.1210-1215. DOI: 10.1109/ICME.2019.00211.

Qing Tian received his Ph.D. degree in computer science from Nanjing University of Aeronautics and Astronautics, Nanjing, in 2016. He is currently an associate professor in the School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing. He was an academic visitor at the University of Manchester, UK, from 2018 to 2019. He is the recipient of the National Ph.D. Scholarship Award of China in 2015, the Best Scientific Paper Award of ICPR in 2016, the Excellent Doctoral Dissertation Award of Jiangsu Province of China in 2017, etc. He has served as a program committee member for several renowned international conferences, such as IJCAI, PRICAI, and IDEAL, and a reviewer for many prestigious international journals and conferences, such as IEEE TPAMI, IEEE TNNLS, IEEE TCYB, IEEE TIFS, ACM TIST, IJCAI, ICDM, and CVPR. His research interests include machine learning, pattern recognition and computer vision. He is a member of CCF.



Chuang Ma received his B.S. degree in computer science from Nanjing University of Information Science and Technology (NUIST) in 2018, Nanjing. He is currently pursuing his Master' degree at NUIST, Nanjing. His research interests include machine learning and pattern recognition.

Feng-Yuan Zhang is currently pursuing his B.S. degree at the Nanjing University of Information Science and Technology, Nanjing. His research interests include machine learning and pattern recognition.



Shun Peng received his B.S. degree in computer science from Nanjing University of Information Science and Technology (NUIST), Nanjing, in 2020. He is currently pursuing his Master's degree at NUIST, Nanjing. His research interests include machine learning and pattern recognition.

Hui Xue received her B.Sc. degree in mathematics from Nanjing Normal University, Nanjing, in 2002. In 2005, she received her M.Sc. degree in mathematics from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing. And she also received her Ph.D. degree in computer application technology at NUAA, Nanjing, in 2008. Since 2009, as a professor, she has been with the School of Computer Science and Engineering at Southeast University, Nanjing. Her research interests include pattern recognition and machine learning. She is a member of CCF.