Neurocomputing 349 (2019) 314–325

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Short communication

Joint Domain Matching and Classification for cross-domain adaptation via ELM☆

Chao Chen, Buyuan Jiang, Zhaowei Cheng, Xinyu Jin*

Institute of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China

ARTICLE INFO

Article history:
Received 15 May 2018
Revised 14 January 2019
Accepted 17 January 2019
Available online 25 January 2019

Communicated by Dr Li Sheng

Keywords:
Domain adaptation
Extreme learning machine
Domain matching
Joint learning model
Feature selection
Output adaptation

ABSTRACT

Recent years, domain adaptation has attracted much attention in the community of machine learning. In this paper, we mainly focus on the tasks of Joint Domain Matching and Classification (*JDMC*) under the framework of extreme learning machine (ELM). Specifically, our JDMC method is formulated by optimizing both the output-adapted transformation and the cross-domain classifier, which allows us to simultaneously (1) align the source domain and target domain in the feature space with correlation alignment, (2) minimize the discrepancy between the source and target domain, measured in terms of both marginal and conditional probability distribution in the mapped feature space, (3) select informative features which behave similarly in both domains for knowledge transfer by imposing $\ell_{2,1}$-norm on the output weights of ELM. In this respect, the proposed JDMC integrates the feature matching, feature selection and classifier design in a unified framework. Besides, an efficient alternative optimization strategy is exploited to solve the joint learning model. To evaluate the effectiveness of the proposed method, extensive experiments on several commonly used domain adaptation datasets are presented, the results show that the proposed method significantly outperforms the non-transfer ELM networks and consistently outperforms several state-of-art domain adaptation methods.

1. Introduction

There is a strong assumption in traditional pattern classification methods that all the data are drawn from the same distribution, which may not always hold in many real world scenarios. For example, in cases where the training samples are difficult or expensive to obtain, or the distribution of the samples changes over time, we have to borrow knowledge from other different but highly related domains. To address these issues, domain adaptation which aims to train a satisfactory classifier with limited target domain samples and sufficient source domain samples, has emerged as a new framework to solve this problem in the past decades, and received more and more attention in recent years. As has been concluded in [1,2], the commonly used domain adaptation approaches can be roughly classified into three categories: (1) feature matching based domain adaptation, (2) instance reweighting based domain adaptation and (3) classifier-based domain adaptation. The feature matching based methods are the most widely used domain adaptation approaches [3–17], which aim to learn a shared feature representation to minimize the distribution discrepancy between the source domain and target domain. These methods can be further distinguished by: (a) the considered class of transformations, which are generally defined as projections [10,12,13] or non-linear transformations [9,11] (b) the types of discrepancy metric, such as Maximum Mean Discrepancy (MMD) [3,4], Kullback Leibler divergence (KL) [14], Central Moment Discrepancy (CMD) [15] or other similarity metric [16,17]. The instance reweighting is another typical strategy for domain adaptation [18–20], which considers that some source instances may not be relevant to the target even in the shared subspace. Therefore it minimizes the distribution differences by reweighting the source samples and then learns from the reweighted instances. The classifier-based domain adaptation represents another independent line of work, which adapts the pretrained source model to the target by regularizing the difference of the parameters between the source and target domain [21–23]. Apart from this, low rank [24–27] and discriminative feature learning [28] were also exploited for domain adaptation in recent work.

In spite of the fact that the above approaches are intuitively designed and effective for many domain adaptation problems, they also have some limitations [27,29]. For the feature matching based methods, they may not be effective for large domain shift

☆ This work was supported by the opening foundation of the State Key Laboratory (No. 2014KF06), and the National Science and Technology Major Project (No. 2013ZX03005013).

* Corresponding author.
E-mail addresses: chench@zju.edu.cn (C. Chen), byjiang@zju.edu.cn (B. Jiang), chengzhaowei@zju.edu.cn (Z. Cheng), jinxy@zju.edu.cn (X. Jin).

https://doi.org/10.1016/j.neucom.2019.01.056
0925-2312/

problems, in which the domain distribution difference cannot be appropriately reduced by cross-domain transformation [30]. Besides, majority of this kind of methods only focus on the data representation, followed by a selection of classifier, neglecting the fact that it would be better to combine the classifier design and the feature matching process into a single paradigm [27]. For the instance reweighting based adaptation methods, they require a strict assumption that the conditional distributions of source and target domain are identical, and several instances in the source domain can be reused for learning in the target domain [29]. The representative classifier based methods can be found in [21,22], which adapt the source classifier to the target by modifying the trained model parameters, but remain the data representation unchanged. Therefore, they are only effective for small domain shift problems [2].

To address the aforementioned limitations, we propose the joint domain matching and classification approach based on the ELM network in this paper. We would like to learn a high-quality domain adaptation ELM classifier using a small number of labeled target domain samples and a large number of source domain samples. On the one hand, to learn a shared representation, our model simultaneously (1) aligns the source domain with the target domain in the feature space, (2) minimizes the marginal and conditional distribution discrepancy in the projected feature space, which can also be seen as label space consistency and (3) learns two coupled projections for the output adaptation of the source and target domain. On the other hand, in contrast with the instance reweighting based methods which need a strict assumption, our domain adaptation ELM is incorporated with $\ell_{2,1}$-norm regularization, which encourages joint feature selection. In this way, the classifier is designed to only select the useful features that behave similarly in both domains for knowledge transfer. Moreover, the domain matching, feature selection and the classifier design are integrated into a unified optimization framework to guarantee an optimal solution. For ease of notation, the joint domain matching and classification transfer ELM is referred to as JDMC.

The contributions of this paper are summarized as follows: (1) we are among the first to integrate the domain matching, feature selection and the classifier design in a unified framework, especially under the framework of ELM. (2) Unlike most of existing works which find the shared subspace by input adaptation, our method exploits output adaptation in both domains. (3) Both the marginal and the conditional distribution discrepancy are minimized, while most of existing works minimize the marginal distribution difference only. (4) The $\ell_{2,1}$-norm regularization is incorporated into the classifier design which encourages our model to select informative features for knowledge transfer. (5) Extensive experiments on several challenging datasets are performed, which demonstrate that our proposed method outperforms the non-transfer ELM by a large margin and almost consistently outperforms several state-of-art domain adaptation methods.

2. Related work

The Extreme learning machine (ELM) first proposed by Huang et al. [31], determining its input weights randomly, now plays an important role in the community of machine learning due to its fast learning speed, satisfactory performance and little human intervention [32]. Therefore, since it was first put forward, various extensions have been proposed to make the original ELM model more efficient and suitable for specific applications, such as semi-supervised and unsupervised ELM [33], weighted ELM (WELM) [34], cost-sensitive ELM [35], online sequential ELM [36], ELM auto-encoder (ELM-AE) [37], and multi-layer ELM [38] etc.

It is not until recently that some researchers have extended the classical ELM to domain adaptation ELM [5,6,23,39–41]. Zhang et al. proposed a domain adaptation ELM to address the sensor drift problem in the E-nose system [40]. In [6], a unified subspace transfer framework based on ELM was proposed, which learns a subspace that jointly minimizes the mean distribution discrepancy (MMD) and maximum margin criterion (MMC). Uzair and Mian [39] proposed a blind domain adaptation ELM with extreme learning machine auto-encoder (ELM-AE), which does not need target domain samples for training. Zhang and Zhang [5] proposed an ELM-based domain adaptation (EDA) for visual knowledge transfer and extended the EDA to multi-view learning. In EDA, the manifold regularization was incorporated into the objective function, and the author minimized the $\ell_{2,1}$-norm of the output weights and training errors simultaneously. Besides, the classifier-based transfer learning ELM has also been proposed in [22,23], which regularized the difference of the source and target parameters. In addition, Salaken et al. [42] summarized all the available literatures in the field of ELM-based transfer learning.

There are also some joint domain adaptation methods that have been studied extensively [3,5,10,30,43], which are closely related to our work. The MMDT [10] jointly optimizes the classifier parameters and the transformation matrix which maps the target features into a new feature space maximally aligned with the source. The TJM [3] performs joint feature matching and instance reweighting for robust domain adaptation. The JCSL [43] jointly learns the new domain-invariant representation as well as the prediction function in the unsupervised setting. The DMM [30] jointly learns the transfer classifier and transferable knowledge (invariant feature representations and unbiased instance weights) in an end-to-end learning paradigm. In contrast with the above approaches which are modeled on the basis of SVM, the EDA [5] is the one most related to our proposal, which learns the ELM classifier as well as the category transformation by minimizing the $\ell_{2,1}$-norm of the output weights and the training error simultaneously, whereas our method also takes into account the domain distribution discrepancy measured by MMD and feature space alignment. All these methods are somewhat similar, but also obviously distinct, to our method. The connections between these existing works and our proposal will be detailedly discussed in Section 4.6.

3. Preliminaries

3.1. A Brief Review of ELM

Considering a supervised learning problem where the training set with N samples and their corresponding targets are given as $\{\mathbf{X}, \mathbf{Y}\} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) | \boldsymbol{x}_i \in \mathbb{R}^d, \boldsymbol{y}_i \in \mathbb{R}^c, i = 1, 2, \ldots, N\}$. Here $\boldsymbol{x}_i \in \mathbb{R}^d$ is the d-dimensional input data and $\boldsymbol{y}_i \in \mathbb{R}^c$ is its associated one-hot label. The ELM networks learn a decision rule with the following two stages. In the first stage, it randomly generates the input weights $\boldsymbol{w}$ and bias $\boldsymbol{b}$, and maps the original data from the input space into the L-dimensional feature space $\boldsymbol{h}(\boldsymbol{x}_i) \in \mathbb{R}^L$, where L is the number of hidden nodes, $\boldsymbol{h}(\boldsymbol{x}_i) = \boldsymbol{g}(\boldsymbol{w}^\top \boldsymbol{x}_i + b)$, and $\boldsymbol{g}(\cdot)$ is the activation function. In this respect, the only free parameter of the ELM is the output weights $\boldsymbol{\beta} \in \mathbb{R}^{L \times c}$. In the second stage, the ELM solves the output weights by minimizing both the prediction errors and the norm of the output weights simultaneously, leads to

$$\begin{cases} \min\limits_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta}) = \dfrac{1}{2}\|\boldsymbol{\beta}\|_F^2 + \dfrac{\lambda}{2}\sum\limits_{i=1}^{N}\|\boldsymbol{\xi}_i\|_2^2 \\ \text{s.t.} \quad h(\boldsymbol{x}_i)\boldsymbol{\beta} = \boldsymbol{y}_i - \boldsymbol{\xi}_i, i = 1, 2, \cdots\cdots, N \end{cases} \tag{1}$$

where $\boldsymbol{\xi}_i$ is the prediction error with respect to the i-th training samples, the first term of the objective function is the regularization term preventing the network from overfitting. By substituting the constrain into the objective function, the problem (1) can be

simplified to such an unconstrain optimization problem:

$$\min_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{\beta}\|_F^2 + \frac{\lambda}{2}\|\mathbf{H}\boldsymbol{\beta} - \mathbf{Y}\|_F^2 \tag{2}$$

where $\mathbf{H} = [\boldsymbol{h}(\boldsymbol{x}_1); \boldsymbol{h}(\boldsymbol{x}_2); \cdots\cdots; \boldsymbol{h}(\boldsymbol{x}_N)] \in \mathbb{R}^{N\times L}$. The optimal solution of $\boldsymbol{\beta}$ can then be analytically determined by setting the derivatives of $\mathcal{L}(\boldsymbol{\beta})$ with respect to $\boldsymbol{\beta}$ to be zero, i.e.

$$\frac{\partial \mathcal{L}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \boldsymbol{\beta} + \lambda \mathbf{H}^\top(\mathbf{H}\boldsymbol{\beta} - \mathbf{Y}) = 0 \tag{3}$$

Then, the output weights $\boldsymbol{\beta}$ can be given as

$$\boldsymbol{\beta} = \left(\mathbf{H}^\top\mathbf{H} + \frac{\mathbf{I}}{\lambda}\right)^{-1}\mathbf{H}^\top\mathbf{Y} \tag{4}$$

Here $\mathbf{I}$ is the identity matrix and λ is the regularization coefficient. With the closed-form solution, the ELM model is remarkably efficient and tends to reach a global optimum.

3.2. Notations and Problem Definitions

We summarize the frequently used notations and definitions as below.

Notations: For a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, let the i-th row of $\mathbf{A}$ denoted by $\mathbf{a}^i$. The Frobenius norm of the matrix $\mathbf{A}$ is defined as

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} \mathrm{a}_{\mathrm{ij}}^2} = \sqrt{\sum_{i=1}^{m}\|\mathbf{a}^i\|_2^2} \tag{5}$$

The $\ell_{2,1}$-norm of a matrix, introduced in [44] firstly as rotation invariant ℓ_1-norm which ensures the row sparsity of a matrix, was widely used for feature selection and structured sparsity regularizer [3,5,45,46]. It is defined as

$$\|\mathbf{A}\|_{2,1} = \sum_{i=1}^{m}\sqrt{\sum_{j=1}^{n}\mathrm{a}_{\mathrm{ij}}^2} = \sum_{i=1}^{m}\|\mathbf{a}^i\|_2 \tag{6}$$

Definition 1 (Domain)**.** A domain $\mathcal{D}$ is composed of a feature space $\mathcal{X} \in \mathbb{R}^d$ and a marginal distribution $\mathcal{P}(\mathbf{X})$. i.e. $\mathcal{D} = \{\mathcal{X}, \mathcal{P}(\mathbf{X})\}$, with $\mathbf{X} \in \mathcal{X}$.

Definition 2 (Task)**.** Given a specific domain $\mathcal{D}$, a task $\mathcal{T}$ is composed of a label space $\mathcal{Y}$ and a classifier $f(\boldsymbol{x})$, i.e. $\mathcal{T} = \{\mathcal{Y}, f(\boldsymbol{x})\}$, where $f(\boldsymbol{x})$ can also be interpreted as the conditional probability distribution $\mathcal{Q}(\mathbf{Y} \mid \mathbf{X})$.

Definition 3 (Domain adaption)**.** Given the source domain $\mathcal{D}_s$ and target domain $\mathcal{D}_t$, in which the data are insufficient to learn a high-quality classification model. Domain adaptation aims to learn a satisfied classifier f_t: $\boldsymbol{x}_t \rightarrowtail \boldsymbol{y}_t$ with low expected target error on $\mathcal{D}_t$, under the assumption that the source and target domain are different but related $\mathcal{P}(\mathbf{X}_s) \neq \mathcal{P}(\mathbf{X}_t)$ and $\mathcal{Q}(\mathbf{Y}_s \mid \mathbf{X}_s) \neq \mathcal{Q}(\mathbf{Y}_t \mid \mathbf{X}_t)$.

To address the cross domain issues, most of existing works assume that there exist a transformation $\mathcal{T}$, such that the new representation of the data can be matched, i.e. $\mathcal{P}_s(\mathcal{T}(\mathbf{X}_s)) \approx \mathcal{P}_t(\mathcal{T}(\mathbf{X}_t))$ and $\mathcal{Q}_s(\mathbf{Y}_s \mid \mathcal{T}(\mathbf{X}_s)) \approx \mathcal{Q}_t(\mathbf{Y}_t \mid \mathcal{T}(\mathbf{X}_t))$. The transformations $\mathcal{T}$ are generally defined as projections [10,12,13] or non-linear transformations [9,11], inferred by minimizing the distribution distance between the source and target domain [4]. Different from the conventional domain adaption methods which perform the domain transformation in the input space, in this paper, we also exploit the output adaptation, which learns the transformation in the label space such that $\mathcal{Q}_s(\mathcal{T}(\mathbf{Y}_s) \mid \mathbf{X}_s) \approx \mathcal{Q}_t(\mathcal{T}(\mathbf{Y}_t) \mid \mathbf{X}_t)$. The output adaptation can also be regarded as the label space consistency constraint.

Theoretical bound of the expected target error. Theoretical studies have been developed for the bound on the target domain generalization performance of a classifier trained in the source domain. As analysed in [47], the bound of the expected target error can be estimated as:

$$\boldsymbol{\xi}_T(h) \leq \boldsymbol{\xi}_S(h) + d_{\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) + C \tag{7}$$

Here h is the learned hypothesis which can also be interpreted as the predictor function. $\boldsymbol{\xi}_T(h)$ and $\boldsymbol{\xi}_S(h)$ are the predict error in the target domain and source domain, respectively. $d_{\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t)$ represents the $\mathcal{H}$-divergence between the source and target domain, which can be minimized by reducing the distribution discrepancy. $C = \boldsymbol{\xi}_T(h^*) + \boldsymbol{\xi}_S(h^*)$ is the joint error on both domains with the ideal hypothesis $h^* = \arg\min_h(\boldsymbol{\xi}_S(h) + \boldsymbol{\xi}_T(h))$, which can be disregarded because it is considered to be negligibly small [48]. The generalization bound theory states that a low target error can be guaranteed if both the source error and the domain distribution discrepancy are small.

4. Proposed method

In this section, we present the proposed ELM based Joint Domain Matching and Classification (JDMC) method and its learning algorithm in detail.

Suppose we have a source domain $\mathcal{D}_s = \{(\boldsymbol{x}_s^1, \boldsymbol{y}_s^1), \ldots, (\boldsymbol{x}_s^m, \boldsymbol{y}_s^m)\}$, and a target domain $\mathcal{D}_t = \{(\boldsymbol{x}_t^1, \boldsymbol{y}_t^1), \ldots, (\boldsymbol{x}_t^n, \boldsymbol{y}_t^n)\}$. Generally, in the supervised domain adaptation setting, n is small and $m \gg n$. With the aim of minimizing both the source error and domain distribution discrepancy in (7), the proposed JDMC method is formulated by optimizing both the two coupled projections ($\mathbf{P_s}$ for source domain and $\mathbf{P_t}$ for target domain) and the output weights $\boldsymbol{\beta}$ jointly, such that (1) the training errors on both the source and target domain are small, (2) the distribution discrepancy between the source and target domain is minimized, (3) the informative features that behave similarly on both domains are selected for knowledge transfer and (4) the adaptation is performed in the label space by the learned two coupled projections. Suppose the prediction function (ELM classifier) be $f = \boldsymbol{\beta}^\top \cdot \boldsymbol{h}(\boldsymbol{x})$, where $\boldsymbol{h(x)}$ is the output of the hidden layer. Then, the general framework of JDMC can be formulated as

$$\begin{aligned}\min_{\boldsymbol{\beta},\mathbf{P_s},\mathbf{P_t}} &\sum_{i=1}^{m}\ell(f(\boldsymbol{x}_s), \mathbf{P_s}\boldsymbol{y}_s) + \sum_{i=1}^{n}\ell(f(\boldsymbol{x}_t), \mathbf{P_t}\boldsymbol{y}_t) + J_{MMD}(\mathbf{X}^s, \mathbf{X}^t)\\ &+\|\boldsymbol{\beta}\|_{2,1} + \Omega(\mathbf{P_s}, \mathbf{P_t})\end{aligned} \tag{8}$$

where the first two terms correspond to the training errors of the source domain and the target domain, respectively. $\mathbf{P_s}\boldsymbol{y}_s$ and $\mathbf{P_t}\boldsymbol{y}_t$ represent the adaptation in the output space for the source and target domain. $J_{MMD}(\mathbf{X}^s, \mathbf{X}^t)$ indicates the domain matching loss measured by Maximum Mean Discrepancy (MMD) [4,6,48], and the last term indicates the regularization regarding to the output transformation matrices. We will interpret each term of (8) in the following subsections.

4.1. Feature space pre-alignment

Let $\mathbf{H}_s \in \mathbb{R}^{n\times L}$ and $\mathbf{H}_t \in \mathbb{R}^{m\times L}$ denote the outputs of the hidden layer corresponding with the source and target domain. Since the input parameters of the ELM networks are randomly initialized, the source features and target features in the hidden layer can be calculated in advance. Therefore, the distribution discrepancy in the feature space can be easily reduced by domain alignment. To achieve feature space pre-alignment which could benefit the subsequent processing, the correlation alignment (CORAL) [12] is considered, which is one of the most popular domain alignment methods due to its frustratingly easy implementation and considerable performance. The aim of the correlation alignment is to minimize the distance between the second order covariance of the source and target features by applying a linear

transformation to the source features $\mathbf{H}_s = \mathbf{H}_s\mathbf{A}$. Suppose $\mathbf{C}_s$ and $\mathbf{C}_t$ denote the covariance matrices of the source and target feature space, i.e. $\mathbf{C}_s = Cov(\mathbf{H}_s) = \mathbf{H}_s^\top \mathbf{J}_n \mathbf{H}_s$ and $\mathbf{C}_t = Cov(\mathbf{H}_t) = \mathbf{H}_t^\top \mathbf{J}_m \mathbf{H}_t$, where $\mathbf{J}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top$ and $\mathbf{J}_m = \mathbf{I}_m - \frac{1}{m}\mathbf{1}_m\mathbf{1}_m^\top$ denote the centralized matrix, $\mathbf{1}$ denotes an all-one column vector. Let $\widetilde{\mathbf{C}}_\mathbf{s}$ denotes the covariance matrices of the transformed source features $\mathbf{H}_s\mathbf{A}$, i.e. $\widetilde{\mathbf{C}}_\mathbf{s} = Cov(\mathbf{H}_s\mathbf{A}) = \mathbf{A}^\top\mathbf{H}_s^\top\mathbf{J}_\mathbf{n}\mathbf{H}_s\mathbf{A} = \mathbf{A}^\top\mathbf{C}_\mathbf{s}\mathbf{A}$. Then, the domain shift between the transformed source and target feature space can be measured by the following function

$$\min_{\mathbf{A}} |\widetilde{\mathbf{C}}_s - \mathbf{C}_t\|_F^2 = \|\mathbf{A}^\top\mathbf{C}_s\mathbf{A} - \mathbf{C}_t\|_F^2 \tag{9}$$

The solution of (9) can be given as

$$\mathbf{A} = \frac{(\mathbf{C}_t + \alpha\mathbf{I})^{\frac{1}{2}}}{(\mathbf{C}_s + \alpha\mathbf{I})^{\frac{1}{2}}} \tag{10}$$

Here $\alpha\mathbf{I}$ is the regularization term which guarantees the full rank of the covariance matrix, where α is set to 1 as recommended in [12]. In this way, the feature space could be easily pre-aligned by applying $\mathbf{H}_s = \mathbf{H}_s\mathbf{A}$.

4.2. Domain matching

Recall that, even though the source and target domain have been pre-aligned in the feature space by CORAL, the distribution divergence, especially the conditional distribution divergence is still significantly large. Intuitively, a natural idea is that the features of two domains $\mathbf{H}_s$ and $\mathbf{H}_t$ could be more similar after being mapped by the output weights $\boldsymbol{\beta}$ [6]. In this respect, it is reasonable to reduce the domain distribution discrepancy between the mapped source and target feature space. Hence, we adopt the Maximum Mean Discrepancy (MMD) as the distance measure, which computes the distance between the empirical expectations of the mapped source and target features. It is formulated as

$$\min_{\boldsymbol{\beta}} \|\frac{1}{m}\sum_{x_i\in\mathcal{X}_s}\boldsymbol{h}_s(\boldsymbol{x}_i)\boldsymbol{\beta} - \frac{1}{n}\sum_{x_j\in\mathcal{X}_t}\boldsymbol{h}_t(\boldsymbol{x}_j)\boldsymbol{\beta}\|_F^2 \tag{11}$$

where $\boldsymbol{h}_s(\boldsymbol{x}_i)$ and $\boldsymbol{h}_t(\boldsymbol{x}_j)$ denote the outputs of the hidden layer in the source and target domain, respectively. Except for minimizing the marginal distribution distance with the MMD criterion, in [4], it has also been proposed to reduce the conditional probability distribution by making the intra-class centroid of two distributions closer. For the domains with c class, $k \in \{1, 2, \ldots, c\}$, we follow their idea to minimize the conditional distribution discrepancy as

$$\min_{\boldsymbol{\beta}} \sum_{k=1}^{c} \|\frac{1}{m^{(k)}}\sum_{x_i\in\mathcal{X}_s^{(k)}}\boldsymbol{h}_s(\boldsymbol{x}_i)\boldsymbol{\beta} - \frac{1}{n^{(k)}}\sum_{x_j\in\mathcal{X}_t^{(k)}}\boldsymbol{h}_t(\boldsymbol{x}_j)\boldsymbol{\beta}\|_F^2 \tag{12}$$

where $m^{(k)}$ and $n^{(k)}$ are the number of samples in the kth class in the source and target domain, respectively. $\mathcal{X}_s^{(k)} = \{x_i | x_i \in \mathcal{D}_s \wedge y(x_i) = k\}$ is the set of samples belonging to class k in the source domain, and $\mathcal{X}_t^{(k)} = \{x_j | x_j \in \mathcal{D}_t \wedge y(x_j) = k\}$ is the set of samples belonging to class k in the target domain. By incorporating (11) into (12), the final formulation of the MMD minimization term can be represented as

$$\min_{\boldsymbol{\beta}} \sum_{k=0}^{c} \|\frac{1}{m^{(k)}}\sum_{x_i\in\mathcal{X}_s^{(k)}}\boldsymbol{h}_s(\boldsymbol{x}_i)\boldsymbol{\beta} - \frac{1}{n^{(k)}}\sum_{x_j\in\mathcal{X}_t^{(k)}}\boldsymbol{h}_t(\boldsymbol{x}_j)\boldsymbol{\beta}\|_F^2 \tag{13}$$

here, $m^{(0)} = m, n^{(0)} = n, \mathcal{X}_s^{(0)} = \mathcal{X}_s, \mathcal{X}_t^{(0)} = \mathcal{X}_t$. (13) can be further written compactly as

$$\min_{\boldsymbol{\beta}} \sum_{k=0}^{c} \left\| \left(\frac{1}{m^{(k)}}\sum_{x_i\in\mathcal{X}_s^{(k)}}\boldsymbol{h}_s(\boldsymbol{x}_i) - \frac{1}{n^{(k)}}\sum_{x_j\in\mathcal{X}_t^{(k)}}\boldsymbol{h}_t(\boldsymbol{x}_j) \right)\boldsymbol{\beta} \right\|_F^2 \tag{14}$$

$$\Rightarrow \min_{\boldsymbol{\beta}} \sum_{k=0}^{c} \|(\boldsymbol{\mu}_s^{(k)} - \boldsymbol{\mu}_t^{(k)})\boldsymbol{\beta}\|_F^2 \tag{15}$$

where $\boldsymbol{\mu}_s^{(0)}$ and $\boldsymbol{\mu}_t^{(0)}$ denote the centroid of the source and target feature space, $\boldsymbol{\mu}_s^{(k)}$ and $\boldsymbol{\mu}_t^{(k)}$ $(k = 1, 2, \ldots, c)$ denote the centroid of the kth class in the source and target feature space. For simplification, we define $\Delta\boldsymbol{\mu}^{(k)} = \boldsymbol{\mu}_s^{(k)} - \boldsymbol{\mu}_t^{(k)}$, then the final distribution discrepancy minimization term can be represented as

$$\min_{\boldsymbol{\beta}} \sum_{k=0}^{c} \|\Delta\boldsymbol{\mu}^{(k)}\boldsymbol{\beta}\|_F^2 \tag{16}$$

4.3. Joint learning model

In this paper, we aim to jointly learn the transformations as well as the cross domain ELM classifier, while minimizing the distribution discrepancy between the source and target domain, and selecting the informative features for knowledge transfer. To achieve this goal, the proposed JDMC model can be formulated as

$$\begin{aligned}\min_{\boldsymbol{\beta},\mathbf{P}_s,\mathbf{P}_t} \quad & \mathcal{L}(\boldsymbol{\beta},\mathbf{P}_s,\mathbf{P}_t) = \frac{1}{2}\|\mathbf{H}_t\boldsymbol{\beta} - \mathbf{P}_t\mathbf{Y}_t\|_F^2 + \frac{\lambda_1}{2}\|\mathbf{H}_s\boldsymbol{\beta} - \mathbf{P}_s\mathbf{Y}_s\|_F^2 \\ & + \frac{\lambda_2}{2}\sum_{k=0}^{c}\|\Delta\boldsymbol{\mu}^{(k)}\boldsymbol{\beta}\|_F^2 + \frac{\lambda_3}{2}\|\boldsymbol{\beta}\|_{2,1} + \frac{\lambda_4}{2}(\|\mathbf{P}_s - \mathbf{I}\|_F^2 + \|\mathbf{P}_t - \mathbf{I}\|_F^2)\end{aligned} \tag{17}$$

where λ_1, λ_2, λ_3 and λ_4 are trade-off parameters to balance the contributions of the four regularizers. $\mathbf{P}_t \in \mathbb{R}^{n\times n}$ and $\mathbf{P}_s \in \mathbb{R}^{m\times m}$ are two transformations applied to the output space. The first two terms tend to learn the adaptive ELM classifier and the cross-domain transformations simultaneously. Specifically, the first term corresponds to the training errors in the target domain, while the second term is the regularization with respect to the source domain training errors. As stated above, the third regularization term encourages small marginal and conditional distribution divergence measured by MMD criterion. Besides, the last term is the regularizer *w.r.t.* the two coupled projections, which controls the output space distortion during transformation.

It is worth noting that the $\ell_{2,1}$-norm instead of the Frobenius norm is imposed on the domain adaptive classifier as a regularizer. Benefiting from the property that the $\ell_{2,1}$-norm regularization guarantees the row sparsity of the output weights $\boldsymbol{\beta}$, our joint learning model tends to select the informative features for knowledge transfer.

4.4. Learning algorithm

As can be seen in (17), our goal is to jointly learn the cross domain classifier $\boldsymbol{\beta}$ as well as the two coupled projections $\mathbf{P}_s$ and $\mathbf{P}_t$. Since there are three free parameters to be solved, this optimization problem cannot be directly solved like problem (2). Therefore, the coordinate descent method, which is an alternative optimization strategy that optimizes one variable while fixing the other free variables is considered. The following three main steps are included.

Step 1. Optimizing on $\boldsymbol{\beta}$: In the first step, we fix the projection matrix as $\mathbf{P}_s = \mathbf{I}_{m\times m}$ and $\mathbf{P}_t = \mathbf{I}_{n\times n}$. Then, the sub-problem $\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t)$ can be solved by setting the derivative of objective function *w.r.t.* $\boldsymbol{\beta}$ to be zero. Then we have

$$\begin{aligned}\frac{\partial\mathcal{L}(\boldsymbol{\beta},\mathbf{P}_s,\mathbf{P}_t)}{\partial\boldsymbol{\beta}} &= \mathbf{H}_t^\top(\mathbf{H}_t\boldsymbol{\beta} - \mathbf{P}_t\mathbf{Y}_t) + \lambda_1\mathbf{H}_s^\top(\mathbf{H}_s\boldsymbol{\beta} - \mathbf{P}_s\mathbf{Y}_s) \\ &\quad + \lambda_2\mathbf{U}\boldsymbol{\beta} + \lambda_3\mathbf{D}\boldsymbol{\beta} = 0\end{aligned} \tag{18}$$

$$\Rightarrow (\mathbf{H}_t^\top \mathbf{H}_t + \lambda_1 \mathbf{H}_s^\top \mathbf{H}_s + \lambda_2 \mathbf{U} + \lambda_3 \mathbf{D})\boldsymbol{\beta} = (\mathbf{H}_t^\top \mathbf{P}_t \mathbf{Y}_t + \lambda_1 \mathbf{H}_s^\top \mathbf{P}_s \mathbf{Y}_s) \tag{19}$$

where $\mathbf{U} \in \mathbb{R}^{L\times L}$ is computed as $\mathbf{U} = \sum_{k=0}^{c} \Delta\boldsymbol{\mu}^{(k)\top} \Delta\boldsymbol{\mu}^{(k)}$. Note that $\|\boldsymbol{\beta}\|_{2,1}$ is a non-smooth function at zero, therefore, we compute its sub-gradient instead [3,45,46]. i.e. $\frac{\partial \|\boldsymbol{\beta}\|_{2,1}}{\partial \boldsymbol{\beta}} = 2\mathbf{D}\boldsymbol{\beta}$, where $\mathbf{D}$ is a diagonal sub-gradient matrix with the ith element as

$$\mathbf{D}_{ii} = \frac{1}{2\|\boldsymbol{\beta}^i\|_2 + \epsilon} \tag{20}$$

Here, $\boldsymbol{\beta}^i$ denotes the ith row of $\boldsymbol{\beta}$, ϵ set as a very small constant to prevent the dividend to be zero. With the fixed matrix $\mathbf{D}$, $\boldsymbol{\beta}$ could be solved according to (19), as

$$\boldsymbol{\beta} = (\mathbf{H}_t^\top \mathbf{H}_t + \lambda_1 \mathbf{H}_s^\top \mathbf{H}_s + \lambda_2 \mathbf{U} + \lambda_3 \mathbf{D})^{-1} \times (\mathbf{H}_t^\top \mathbf{P}_t \mathbf{Y}_t + \lambda_1 \mathbf{H}_s^\top \mathbf{P}_s \mathbf{Y}_s) \tag{21}$$

Recall that the sub-gradient matrix $\mathbf{D}$ is dependent on the unsolved parameters $\boldsymbol{\beta}$. Therefore, we employ an alternative optimization strategy to solve $\boldsymbol{\beta}$ according to (21) and (20). In each iteration, only one parameter is updated with the other one fixed. The algorithm is summarized in Algorithm 1. It is worth noting

Algorithm 1: An efficient iterative algorithm to solve $\boldsymbol{\beta}$.

Input: $\mathbf{H}_s, \mathbf{H}_t, \mathbf{P}_s, \mathbf{P}_t, \mathbf{Y}_s, \mathbf{Y}_t$, regularization parameters $\{\lambda_1, \lambda_2, \lambda_3\}$
Output: $\boldsymbol{\beta}$
Set t=0. Initialize $\mathbf{D}^0$ as an identity matrix $\mathbf{I}_{L\times L}$;
Compute matrix $\mathbf{U} = \sum_{k=0}^{c} \Delta\boldsymbol{\mu}^{(k)\top} \Delta\boldsymbol{\mu}^{(k)}$;
repeat
  Update $\boldsymbol{\beta}^{t+1}$ according to (21)
  Update $\mathbf{D}^{t+1}$ according to (20)
  $t = t + 1$
until *Converges*;

that the iterative procedures will be terminated once the number of iterations reaches T_{max} or the output weight $\boldsymbol{\beta}$ tends to convergence. The convergency of Algorithm 1 can be easily proved similar to [3,45].

Step 2. Optimizing on $\mathbf{P}_s$: With the fixed $\boldsymbol{\beta}$ and $\mathbf{P}_t$, the sub-problem $\mathbf{P}_s^* = \arg\min_{\mathbf{P}_s} \mathcal{L}(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P_t})$ can be easily solved by taking the derivative of (17) with respect to $\mathbf{P}_s$ to be zero. We have

$$\frac{\partial \mathcal{L}(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t)}{\partial \mathbf{P}_s} = (\mathbf{P}_s \mathbf{Y}_s - \mathbf{H}_s\boldsymbol{\beta})\mathbf{Y}_s^\top + \lambda_4(\mathbf{P}_s - \mathbf{I}) = 0 \tag{22}$$

$$\Rightarrow \mathbf{P}_s(\mathbf{Y}_s\mathbf{Y}_s^\top + \lambda_4\mathbf{I}) = \mathbf{H}_s\boldsymbol{\beta}\mathbf{Y}_s^\top + \lambda_4\mathbf{I} \tag{23}$$

which leads to

$$\mathbf{P}_s = (\mathbf{H}_s\boldsymbol{\beta}\mathbf{Y}_s^\top + \lambda_4\mathbf{I})(\mathbf{Y}_s\mathbf{Y}_s^\top + \lambda_4\mathbf{I})^{-1} \tag{24}$$

Step 3. Optimizing on $\mathbf{P}_t$: Similar to step 2, the sub-problem $\mathbf{P}_t^* = \arg\min_{\mathbf{P}_t} \mathcal{L}(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t)$ can be similarly solved by taking the derivative of (17) with respect to $\mathbf{P}_t$ to be zero. i.e.

$$\frac{\partial \mathcal{L}(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t)}{\partial \mathbf{P}_t} = (\mathbf{P}_t \mathbf{Y}_t - \mathbf{H}_t\boldsymbol{\beta})\mathbf{Y}_t^\top + \lambda_4(\mathbf{P}_t - \mathbf{I}) = 0 \tag{25}$$

$$\Rightarrow \mathbf{P}_t(\mathbf{Y}_t\mathbf{Y}_t^\top + \lambda_4\mathbf{I}) = \mathbf{H}_t\boldsymbol{\beta}\mathbf{Y}_t^\top + \lambda_4\mathbf{I} \tag{26}$$

leads to

$$\mathbf{P}_t = (\mathbf{H}_t\boldsymbol{\beta}\mathbf{Y}_t^\top + \lambda_4\mathbf{I})(\mathbf{Y}_t\mathbf{Y}_t^\top + \lambda_4\mathbf{I})^{-1} \tag{27}$$

Algorithm 2: Learning Algorithm of the JDMC Method.

Input: Training samples of source and target domain $\mathcal{D}_s = (\mathbf{X}_s, \mathbf{Y}_s), \mathcal{D}_t = (\mathbf{X}_t, \mathbf{Y}_t)$; Regularization parameters $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4\}$; Number of hidden nodes L.
Output: Model parameters $\{\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t\}$.
1. Initialize the networks with randomly selected input weights $\boldsymbol{w}$ and bias $\boldsymbol{b}$;
2. Calculate the hidden layer output matrix $\mathbf{H}_s$ and $\mathbf{H}_t$ with the randomly initialized input parameters;
3. Calculate the transformation matrix $\mathbf{A}$ according to (10), and then align the feature space between the source domain and target domain by $\mathbf{H}_s = \mathbf{H}_s\mathbf{A}$;
4. Set t=0. Initialize $\mathbf{P}_s = \mathbf{I}_{m\times m}$, and $\mathbf{P}_t = \mathbf{I}_{n\times n}$;
repeat
  Update $\boldsymbol{\beta}^{t+1}$ according to Algorithm 1 ;
  Update $\mathbf{P}_s^{t+1}$ according to (24) ;
  Update $\mathbf{P}_t^{t+1}$ according to (27) ;
  $t = t + 1$;
until *Converges*;

The overall learning algorithm is summarized in Algorithm 2. With the randomly initialized input parameters, the hidden layer outputs of the source and target ELM model, which are represented as $\mathbf{H}_s$ and $\mathbf{H}_t$, could be calculated beforehand. Then, we pre-align the two domains using the correlation alignment by applying $\mathbf{H}_s = \mathbf{H}_s\mathbf{A}$ according to (10). After that, in each iteration, we update $\boldsymbol{\beta}$ with current $\mathbf{P}_s$ and $\mathbf{P}_t$, then, update $\mathbf{P}_s$ with the current calculated $\boldsymbol{\beta}$ and $\mathbf{P}_t$, lastly update $\mathbf{P}_t$ with the current calculated $\boldsymbol{\beta}$ and $\mathbf{P}_s$. Note that all of the sub-problems involved are convex (see Section 4.5 for detailed proof), i.e. the joint objective function reaches the optimum in each iteration with the closed-form solution. Hence, the learning algorithm will converge to a minimal after limited number of iterations. The convergency of Algorithm 2 will be analysed in Section 4.5.

4.5. Convergence analysis

In this section, we provide a brief convergence analysis of Algorithm 2. We start by giving the following theorem.

Theorem 1. *All the three sub-problems involved in the joint objective function (17) are convex.*

Proof. For the sub-problem 1: $\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t)$, when the parameters $\{\mathbf{P}_s, \mathbf{P}_t\}$ are fixed, it is straightforward to show that the objection function is convex with respect to $\boldsymbol{\beta}$. According to (18), the second order derivative of the objective function with respect to $\boldsymbol{\beta}$ can be easily calculated as:

$$\frac{\partial \mathcal{L}^2(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t)}{\partial \boldsymbol{\beta}^2} = \mathbf{H}^\top\mathbf{H} + \lambda_1\mathbf{H}_s^\top\mathbf{H}_s + \lambda_2\mathbf{U} + \lambda_3\mathbf{D} > 0 \tag{28}$$

since $\mathbf{U} = \sum_{k=0}^{c} \Delta\boldsymbol{\mu}^{(k)\top} \Delta\boldsymbol{\mu}^{(k)}$ and $\mathbf{D}$ defined in (20) is a diagonal matrix, therefore, all the four terms in (28) are positive definite. i.e. the second order derivative of the objective function with respect to $\boldsymbol{\beta}$ is positive definite, hence the joint objective function (17) is convex to $\boldsymbol{\beta}$.

For the sub-problem 2: $\mathbf{P}_s^* = \arg\min_{\mathbf{P}_s} \mathcal{L}(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t)$, when the parameters $\{\boldsymbol{\beta}, \mathbf{P}_t\}$ are fixed, as can be seen in (22), the second order derivative of the objective function with respect to $\mathbf{P}_s$ can be given as:

$$\frac{\partial \mathcal{L}^2(\boldsymbol{\beta}, \mathbf{P}_s, \mathbf{P}_t)}{\partial \mathbf{P}_s^2} = \mathbf{Y}_s^\top\mathbf{Y}_s + \lambda_4\mathbf{I} > 0 \tag{29}$$

it is clear that the second order derivative of the objective function with respect to $\mathbf{P}_s$ is positive definite, i.e. the sub-problem 2 is convex. Similar to the sub-problem 2, the sub-problem 3 is also convex. Then, Theorem 1 is proven. □

Theorem 2. *The objective function in (17) is monotonically non-increasing in each iterations in Algorithm 2.*

Proof. As stated above, all the three sub-problems involved in the joint objective function are convex. Therefore, the objective function (17) can be minimized at each iteration. Then, we have the following three claims.

Claim 1: $\mathcal{L}(\boldsymbol{\beta}^{t+1}, \mathbf{P}_s^t, \mathbf{P}_t^t) \leq \mathcal{L}(\boldsymbol{\beta}^t, \mathbf{P}_s^t, \mathbf{P}_t^t)$

Claim 2: $\mathcal{L}(\boldsymbol{\beta}^{t+1}, \mathbf{P}_s^{t+1}, \mathbf{P}_t^t) \leq \mathcal{L}(\boldsymbol{\beta}^{t+1}, \mathbf{P}_s^t, \mathbf{P}_t^t)$

Claim 3: $\mathcal{L}(\boldsymbol{\beta}^{t+1}, \mathbf{P}_s^{t+1}, \mathbf{P}_t^{t+1}) \leq \mathcal{L}(\boldsymbol{\beta}^{t+1}, \mathbf{P}_s^{t+1}, \mathbf{P}_t^t)$

Combining the three claims, we have

$$\begin{aligned} &\mathcal{L}(\boldsymbol{\beta}^{t+1}, \mathbf{P}_s^{t+1}, \mathbf{P}_t^{t+1}) \leq \mathcal{L}(\boldsymbol{\beta}^{t+1}, \mathbf{P}_s^{t+1}, \mathbf{P}_t^t) \\ &\leq \mathcal{L}(\boldsymbol{\beta}^{t+1}, \mathbf{P}_s^t, \mathbf{P}_t^t) \leq \mathcal{L}(\boldsymbol{\beta}^t, \mathbf{P}_s^t, \mathbf{P}_t^t) \end{aligned} \tag{30}$$

Then, Theorem 2 is proven. Note that we also conduct empirical convergency evaluation in Section 5.5. □

4.6. Connections to existing works

In this section, we analysis the connections between our proposal and other highly related methods.

Connections to domain alignment methods. The well-known Subspace Alignment (SA) [13] aims to learn a linear transformations that minimizes the Frobenius norm of the difference between the subspaces of source and target domain, while the Correlation Alignment (CORAL) [12] minimizes the domain shifts by aligning the second order statistics of the source and target distributions. Even though the subspace bias or the second order statistics can be effectively aligned by these domain alignment methods, the source and target distributions in the aligned subspace can still be different [49]. In our JDMC, we not only align the second order statistics in the feature space but also match the marginal and conditional probability distributions in the mapped feature space, which guarantees lower domain distribution discrepancy. Besides, we integrate the domain matching and classifier design into a unified framework.

Connections to transform-based methods. Most existing transform-based domain adaption methods perform the domain transformation in the input space [3,10,30], while our approach exploits the adapted transformation in the output space, such that $\mathcal{Q}_s(\mathcal{T}(\mathcal{Y}_s) \mid \mathcal{X}_s) \approx \mathcal{Q}_t(\mathcal{T}(\mathcal{Y}_t) \mid \mathcal{X}_t)$. With the output adaptation, the joint learning model is proved to be joint convex with respect to each model parameter. Therefore, the algorithm is guaranteed to converge to the optimum with the closed-form solution, while many other joint learning models with the input adaptation [3,10,30] do not have this property.

Connections to joint learning methods. To the best of our knowledge, the works most related to our JDMC are [3,5,10,30,43]. All these methods integrate more than two strategies for cross-domain adaptation. The detailed comparison is illustrated in Table 1. As can be seen, our JDMC simultaneously exploits (1) subspace alignment in the feature space, (2) distribution matching of both marginal and conditional distribution in the mapped feature space, (3) output adaptation, (4) informative feature selection and (5) classifier design in a unified framework of ELM.

Table 1
Comparison between most related works.

Methods	MMDT	TJM	JCSL	DMM	EDA	JDMC
Subspace alignment	o	o	✓	o	o	✓
Marginal adaptation	o	✓	o	✓	o	✓
Conditional adaptation	o	✓	o	o	o	✓
Classifier design	✓	o	✓	✓	✓	✓
Input adaptation	✓	✓	o	✓	o	o
Output adaptation	o	o	o	o	✓	✓
Feature selection	o	o	o	o	✓	✓
Instance reweighting	o	✓	o	✓	o	o
ELM framework	o	o	o	o	✓	✓

✓ Denotes this strategy is considered in the method, and o denotes this strategy is not considered in the method.

5. Experiments

In this section, we evaluate our proposed JDMC method on two challenging real-world datasets. We start by introducing the datasets as well as baseline approaches, then follow by discussing the results compared with various baselines, and finish by providing parameter sensitivity analysis and convergency evaluation. The source code of our implementation has been released online.[1]

5.1. Datasets and setup

Since the ELM is also a powerful regressor [32], the proposed JDMC is supposed to be applicable for both classification and regression tasks [50,51] (when JDMC is used for regression tasks, the larger λ_4 is recommended to prevent the label space over distorted). However, following the setup of [3,5,10,43], we only evaluate our approach on classification tasks. Two types of domain adaptation problems are considered: object recognition and text categorization. A summary of the properties of each domain considered in our experiments is provided in Table 2.

Caltech-Office dataset. This dataset [9] consists of *Office* [8] and *Caltech-256* [52] datasets. It contains images from four different domains: *Amazon* (product images download form amazon.com), *Webcam* (low-resolution images taken by a webcam), *Dslr* (high-resolution images taken by a digital SLR camera) and *Caltech*. Ten common categories are extracted from all four domains with each category consisting of 8–151 samples, and 2533 images in total. Several factors (such as image resolution, lighting condition, noise, background and viewpoint) cause the shift of each domain. Fig. 1 highlights the differences among these domains with several selected images from categories of keyboards and headphones. During the experiments, the SURF-BoW image features (*SURF* in short) provided by Hoffman et al. [9] are considered, which encode the images with 800-bin histograms with the codebook trained from a subset of Amazon images using SURF descriptors [53]. These histograms are then normalized to be zero means and unit variance in each dimension.

Multilingual reuters collection dataset. This dataset[2] [54], which is collected by sampling from the Reuters RCV1 and RCV2 collections, contains feature characteristics of 111,740 documents originally written in five different languages and their translations (i.e. English, French, German, Italian, and Spanish), over a common set of 6 categories (i.e. C15, CCAT, E21, ECAT, GCAT, and M11). Documents belonging to more than one of the 6 categories are assigned the label of their smallest category. Therefore, there are 12–30K documents per language, and 11-34K documents per category. All documents are represented as a bag of words and then the TF-IDF features are extracted.

Baselines We compare the results with the following baselines and competing methods that are well adapted for domain shift scenarios:

[1] https://github.com/chenchao666/JDMC.

[2] http://ama.liglab.fr/~amini/DataSets/Classification/Multiview/ReutersMutliLingualMultiView.htm.

Table 2
Summary of the domains used in the experiments.

Problem	Domains	Dataset	# Samples	# Features	# Classes	Abbr.
Objects	Amazon	Office	958	800	10	A
	Webcam	Office	295	800	10	W
	DSLR	Office	157	800	10	D
	Caltech	Caltech-256	1123	800	10	C
Texts	English	Multilingual	18,758	11,547	6	EN
	French	Multilingual	26,648	11,547	6	FR
	German	Multilingual	29,953	11,547	6	GR
	Italian	Multilingual	24,039	11,547	6	IT
	Spanish	Multilingual	12,342	11,547	6	SP

Fig. 1. Selected images from *Office-Caltech* dataset and Caltech-256 dataset. *Amazon, Dslr* and *Webcam* are selected from *Office* dataset while *Caltech* is selected from *Caltech-256* dataset. It is obvious that domain shifts are significant across different domains. (Best viewed in color.)

- **SVM$_s$:** It utilizes sufficient labeled data from source domain to train a standard support vector machine.
- **SVM$_t$**:It utilizes limited labeled data from target domain to train a standard support vector machine.
- **ELM$_s$**:It utilizes sufficient labeled data from source domain to train a extreme learning machine.
- **ELM$_t$**: It utilizes limited labeled data from target domain to train a extreme learning machine.
- **GFK** [9]: It integrates an infinite number of subspaces that characterize changes in geometric and statistical properties from the source to the target domain. We apply it to both source and target domain and use one-Nearest Neighbor as classifier.
- **MMDT** [7,10]: It jointly learns the cross-domain classifier and linear transformation that maps features from the target domain into the source domain.
- **CDLS** [55]: It is able to identify representative cross-domain samples, including the unlabeled ones in the target domain, for performing adaptation.

Regarding the optimal hyper-parameters, they are determined in two steps by applying multiple experiments using grid search strategy. Firstly, we search the approximate range of all the hyper-parameters in the range of $\{0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10,000\}$, after that the candidate parameters can be determined accordingly. Specifically, λ_1, λ_2 are selected from {0.01, 0.05, 0.1, 0.5, 1}, λ_3 is selected from {1, 5, 10, 20, 30, 40, 50, 100}, and λ_4 is chosen in the range of {1, 10, 50, 100, 200, 500, 1000}. During experiments, the hyper-parameters are searched in the range of these values and fixed as the optimal one to get the best results, and the number of hidden nodes is set as $L = 600$ in all experiments.

5.2. Cross-domain object recognition

For our first experiment, we use the *Caltech-Office* domain adaptation benchmark dataset to evaluate our method on the real world computer vision adaptation task.

5.2.1. Experiment setup

Following the setup of [8–10], the number of selected labeled source samples per class for *amazon, webcam, dslr* and *caltech* is 20, 8, 8, and 8, respectively. Instead, when they are used as target domain, 3 labeled target samples are selected. We use the same 20 random train/test splits download from the website[3] provided by the authors [10] for fair comparison and report averaged results across them.

For our method, the optimal parameters are searched in the range of candidate parameters, and the best results are reported. For the other baseline methods, we use the recommended parameters.

5.2.2. Results

As listed in Table 3, we report the mean and standard deviation of classification accuracies for all methods on the *Office-Caltech* dataset. Note that the results in the same column are based on the same 20 random trials for fair comparison. As can be seen, our proposed method shows competitive performance and outperforms all the other methods in 8 out of the 12 individual domain shifts. It is worth noting that our JDMC significantly outperforms the other competing methods when *amazon* is used as source or target domain. We believe the reason is that the domain shift between *amazon* and *dslr* pair or *amazon* and *webcam* pair is more significant than other domain shifts, since the performance discrepancy between ELM$_s$ and ELM$_t$ is larger than other pairs like *webcam* and *dslr*. Similarly, we observe that on the A$\rightarrow$C, D$\rightarrow$W and W$\rightarrow$D domain shifts, our JDMC performs somehow unsatisfactory. We believe it must be caused by the small domain divergence between these domain shifts, as the SVM$_s$ and ELM$_s$ which are only trained on the source domain achieve the highest accuracy. Therefore, we can draw the conclusion that our proposed JDMC is more effective for large domain discrepancy problems.

We also visualize the effectiveness of the proposed JDMC via the confusion matrix. Fig. 2 illustrates the confusion matrices of

[3] https://people.eecs.berkeley.edu/~jhoffman/domainadapt/.

Table 3
Recognition accuracies (%) on the Caltech-Office datasets with SURF feature.

Method	A→C	A→D	A→W	C→A	C→D	C→W	D→A	D→C	D→W	W→A	W→C	W→D
SVM_S	**38.6 ± 0.4**	33.4 ± 1.3	34.8 ± 0.8	38.5 ± 0.6	33.9 ± 1.0	30.2 ± 1.0	36.4 ± 0.5	32.8 ± 0.3	*76.6 ± 0.8*	34.1 ± 0.6	29.6 ± 0.6	67.9 ± 0.7
SVM_T	34.2 ± 0.6	55.5 ± 0.8	63.1 ± 0.8	47.0 ± 1.1	55.3 ± 1.1	59.4 ± 1.4	46.5 ± 1.0	33.4 ± 0.6	60.3 ± 1.2	48.5 ± 0.9	31.1 ± 0.8	53.5 ± 1.0
ELM_S	*36.8 ± 0.4*	31.2 ± 1.2	31.0 ± 1.1	38.1 ± 0.7	35.2 ± 1.0	30.3 ± 1.3	36.5 ± 0.6	30.7 ± 0.5	**78.2 ± 0.5**	32.7 ± 0.7	29.1 ± 0.5	**72.8 ± 0.9**
ELM_T	33.2 ± 0.7	54.5 ± 1.0	*65.5 ± 1.1*	48.8 ± 0.9	56.6 ± 0.8	*64.8 ± 1.4*	48.6 ± 0.9	34.0 ± 0.7	65.9 ± 0.8	*49.9 ± 1.0*	31.4 ± 0.9	57.6 ± 0.8
GFK	*36.0 ± 0.5*	50.7 ± 0.8	58.6 ± 1.0	44.7 ± 0.8	**57.7 ± 1.1**	63.7 ± 0.8	45.7 ± 0.6	32.9 ± 0.5	*76.5 ± 0.5*	44.1 ± 0.4	31.1 ± 0.6	*70.5 ± 0.7*
MMDT	*36.4 ± 0.8*	*56.7 ± 1.3*	64.6 ± 1.2	*49.4 ± 0.8*	56.5 ± 0.9	63.8 ± 1.1	46.9 ± 1.0	34.1 ± 0.8	74.1 ± 0.8	47.7 ± 0.9	32.2 ± 0.8	64.0 ± 0.7
CDLS	28.7 ± 1.0	54.4 ± 1.3	60.5 ± 1.1	41.0 ± 1.0	53.2 ± 1.1	61.6 ± 0.9	*49.1 ± 0.8*	*35.7 ± 0.6*	75.1 ± 0.8	*49.8 ± 0.7*	**34.6 ± 0.6**	64.0 ± 0.7
JDMC	*36.1 ± 0.75*	**58.5 ± 0.7**	**67.6 ± 0.9**	**52.4 ± 0.9**	**59.5 ± 1.0**	**65.9 ± 1.0**	**52.0 ± 0.8**	**36.4 ± 0.6**	67.8 ± 1.0	**53.3 ± 0.7**	*33.6 ± 0.9*	60.4 ± 1.0

Bold indicates the best result for each domain split. Italic indicates the group of results that are close to the best performing result. (A: Amazon, C: Caltech, D: DSLR and W: Webcam).

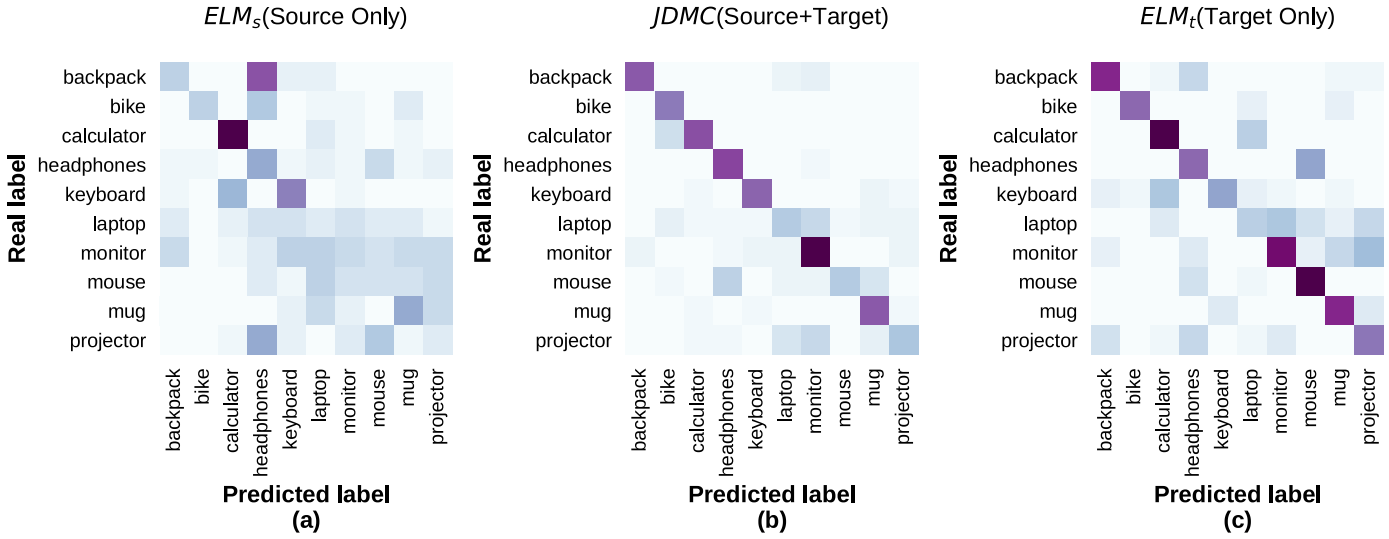

Fig. 2. Confusion matrices of the *amazon* → *webcam* domain adaptation experiment. Left: ELM model trained with source domain only. Middle: our proposed JDMC method trained with source and target domain. Right: ELM model trained with target domain only.

ELM_s, JDMC and ELM_t on *amazon* → *webcam* domain shift experiment. Through the confusion matrix of ELM_s, which is trained with 20 labeled source samples per class, we find that the source only model is heavily confused about several classes. It also reveals the large domain shift between *amazon* and *webcam* and gives explanation for the performance discrepancy between ELM_s and ELM_t. On the other hand, the confusion matrix of ELM_t, which trained with 3 labeled target samples per class, is also somewhat confused due to few labeled training samples. In contrast, as can be seen in Fig. 2(b), the off-diagonal elements of the confusion matrix are close to zero, which demonstrates that our JDMC can effectively transfer the source domain information into the target to train a high-quality cross-domain classifier.

5.3. Cross-domain text categorization

For the second experiment, we utilize the Multilingual Reuters Collection dataset to evaluate our method on the text categorization task.

5.3.1. Experiment setup

In the text dataset, documents written in different languages can be viewed as different domains. We take *Spanish* as target domain, and other four languages (*English, French, German* and *Italian*) as individual source domain. Therefore, there are four domain shifts in total, they are EN → SP, FR → SP, GR → SP and IT → SP, respectively. For each category, we randomly sample 100 labeled training documents from source domain and m labeled training documents from target domain, where $m = 5$, 10, 15 and 20, respectively. And the remaining documents in the target domain are used as the test set.[4] Note that the dimensions of the original TF-IDF features are up to 11,547, in order to fairly compare our method with other competing methods, we perform principal components analysis[5] for dimension reduction and the dimensions after PCA are 40. The parameters are determined in the same way as the first experiment.

5.3.2. Results

We give the box plot of all methods on the Multilingual Reuters Collection dataset when $m = 10$ and 20 in Fig. 3 except SVM_s and ELM_s, since these two methods perform much worse than the other methods. It is obvious that our proposed JDMC method consistently outperforms the competing methods under both settings. Compared with the non-transfer ELM, the performance improvement is nearly 10%. It is also interesting to note that GKF works even worse than ELM_t and SVM_t. A possible explanation is that GFK is put forward for unsupervised domain adaptation without utilizing the label information of target training samples.

We also plot means and standard deviations of all methods over different number of labeled target samples (5, 10, 15 and 20, respectively) in Fig. 4. From the figure, it can be seen that the performance of all methods is improved with the increase of the number of labeled target samples, and our JDMC method performs best

[4] The splits we used can be downloaded from https://github.com/BoyuanJiang/PTELM/tree/master/DataSplits.

[5] The PCA uses randomized singular value decomposition algorithm as SVD solver for efficiency.

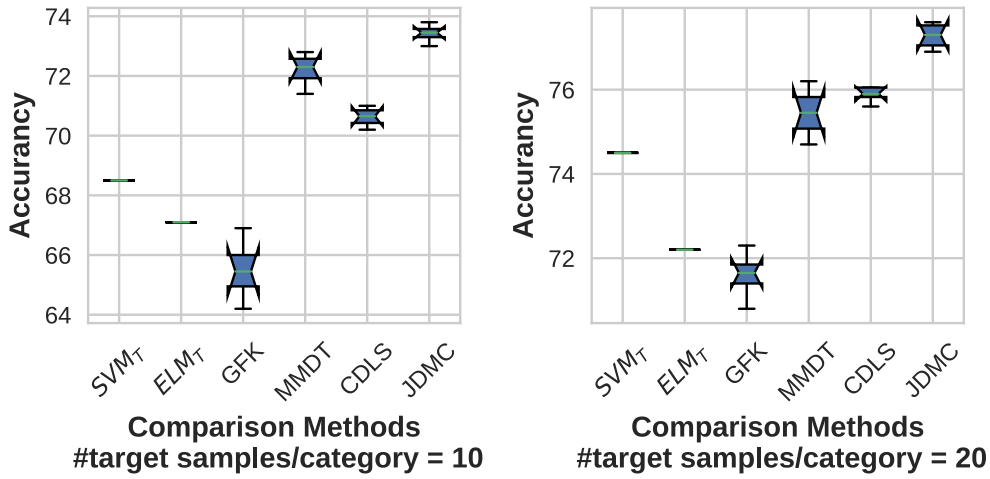

Fig. 3. Box plot illustration of different methods on cross-domain text categorization. Left:we choose 10 labeled target samples per category. Right: we choose 20 labeled target samples per category.

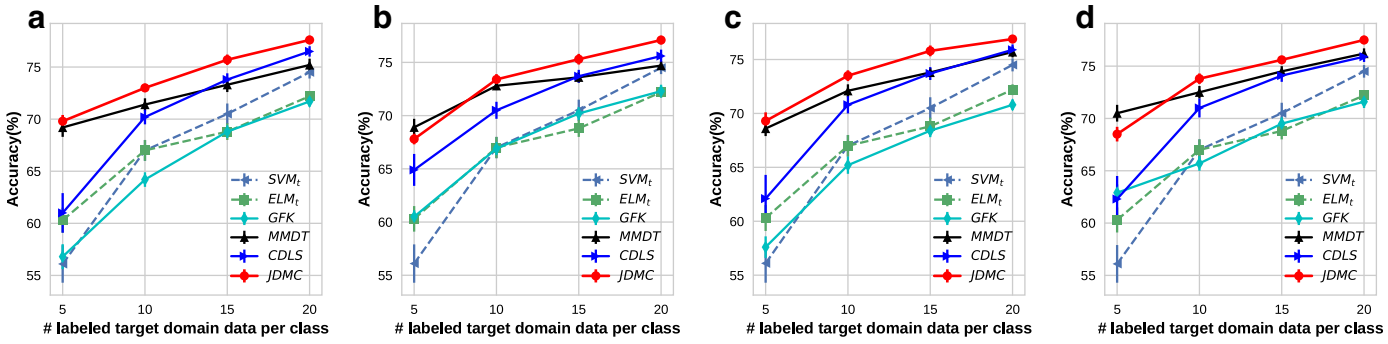

Fig. 4. Classification accuracies of all methods with varied labeled target data per class (i.e. $m = 5$, 10, 15 and 20) on the Multilingual Reuters Collection dataset. Note that *Spanish* is considered as target domain, while the source domains are selected from (a) *English*, (b) *French*, (c) *German* and (d) *Italian*, respectively.

in most cases. Note that the MMDT performs slightly better than our method among two domain shifts, and much better than other baselines when $m = 5$, which demonstrates that MMDT is more suitable when very limited number of labeled target samples are available. Besides, another key insight from the figure is that our method is more stable than the competing methods with lower standard deviations.

5.4. Parameter sensitivity

In this section, we conduct empirical parameter sensitivity analysis of four regularization parameters λ_1, λ_2, λ_3 and λ_4 involved in our method. Due to space limitation, only two domain shifts, i.e. the *amazon* $\rightarrow$ *webcam* from *Office-Caltech* dataset and *IT* $\rightarrow$ *SP* from the Multilingual Reuters Collection dataset are selected for sensitivity analysis. The initial parameters are chosen as $\lambda_1 = 0.1$, $\lambda_2 = 1$, $\lambda_3 = 20$ and $\lambda_4 = 100$, each time, only one parameter is allowed to change with the other parameters fixed. The results are shown as Fig. 5 and we give a brief analysis here. For λ_1, it is used for balancing the contributions of source and target domain. When λ_1 is smaller than 1, the model learns more from the target domain. On the contrary, when λ_1 is larger than 1, the target domain counts more. Since the number of samples in the source domain is always significantly larger than the number of samples in the target domain, the reasonable λ_1 should be smaller than 1, otherwise the target domain information may be "wash out". Therefore, a reasonable value of λ_1 should be $\lambda_1 \in [0.01, 1]$, which is consistent with the experimental results shown in Fig. 5(a). λ_2 corresponds to the penalty term which guarantees small cross-domain distribution divergence. As shown in Fig. 5(b), a suitable value should be $\lambda_2 \in [0.01, 5]$. As can be seen in Fig. 5(c), the λ_3 is the most sensitive parameter in the JDMC method. An inappropriate value of λ_3 will lead to non-convergence of the algorithm. The reasonable choice can be $\lambda_3 \in [10, 30]$. The λ_4 controls the distortions of the transformation. When $\lambda_4 \rightarrow \infty$, the output transformation matrices $\mathbf{P}_s$ and $\mathbf{P}_t$ will converge to identity matrix, which corresponds to the non-output-adaptation situation. When the λ_4 is too small, the label space could be over distorted. Hence, a reasonable value should be $\lambda_4 \in [10, 1000]$. Besides, in order to reflect the effects of the pre-alignment step, we also give the performance without CORAL alignment (dashed line) as contrast. As can be seen, the CORAL pre-alignment can also improve the transfer performance evidently.

5.5. Convergency evaluation

The convergency of the JDMC method has been proved theoretically in Section 4.5, which demonstrates that the objective function is joint convex with respect to $\boldsymbol{\beta}$, $\mathbf{P}_s$ and $\mathbf{P}_t$. Here, we empirically evaluate the convergency performance of the JDMC. In particular, the classification accuracy as well as the variation $\|\boldsymbol{\beta}^{t+1} - \boldsymbol{\beta}^t\|_F^2$ are concerned as the number of iterations increases. As can be seen in Fig. 6, results over two domain shifts are shown, which indicates that both the classification accuracy and the variation $\|\boldsymbol{\beta}^{t+1} - \boldsymbol{\beta}^t\|_F^2$ converges in a limited number of iterations.

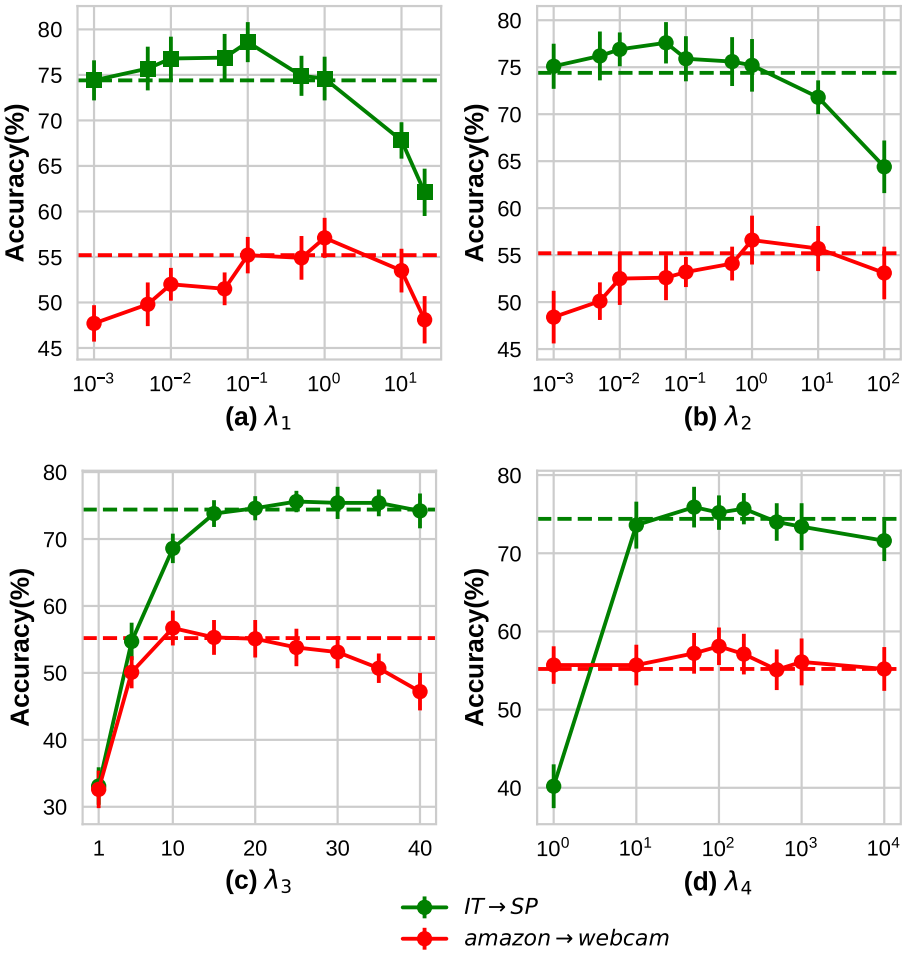

Fig. 5. Parameter Sensitivity. The sensitivity of the regularization parameters λ_1, λ_2, λ_3, λ_4 are evaluated on *EN* → *SP* and *amazon* → *webcam* domain shifts. The dashed line represents the performance of the JDMC without CORAL Alignment in the feature space, which reflects the influences of the pre-alignment step. Note that the standard deviations illustrated in the figure have been amplified for the same scale, since the original value is too small to be seen in the figure.

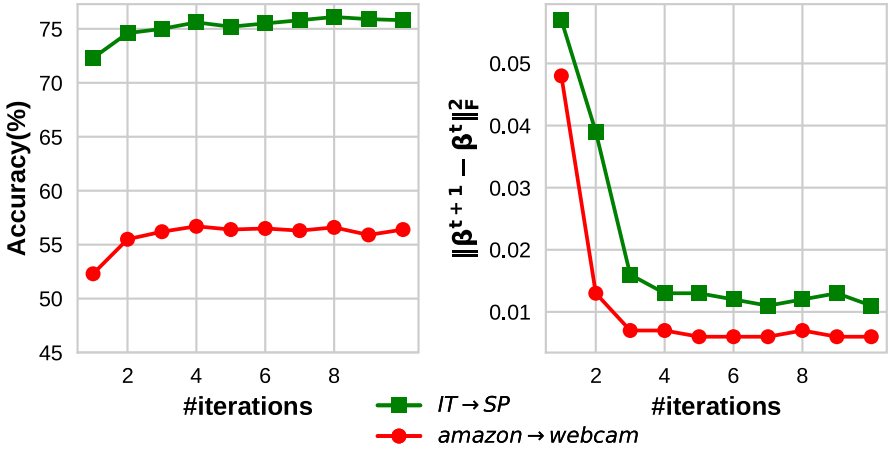

Fig. 6. Convergency evaluation. We empirically evaluate the convergency property of JDMC on *EN* → *SP* and *amazon* → *webcam* domain shifts.

6. Conclusion

In this paper, we presented a novel approach for joint domain adaptation under the ELM framework, which explicitly learns the cross-domain classifier and output adaptation transformations jointly. To reduce the cross-domain distribution discrepancy, we aligned the source and target domain with the correlation alignment, and then minimized the marginal and conditional distribution discrepancy measured by MMD criterion. Besides, in order to select informative features for knowledge transfer, the $\ell_{2,1}$-norm was imposed on the output weights for structured sparsity. What is more, an effective alternative optimization method was introduced to jointly learn the projection matrix and the model parameters. Extensive experiments on several challenging datasets showed the effectiveness of the proposed JDMC compared with the non-transfer ELM and other state-of-art methods.

References

[1] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[2] G. Csurka, Domain adaptation for visual applications: a comprehensive survey, CoRR (2017). abs/1702.05374
[3] M. Long, J. Wang, G. Ding, J. Sun, P.S. Yu, Transfer joint matching for unsupervised domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1410–1417.
[4] M. Long, J. Wang, G. Ding, S.J. Pan, S.Y. Philip, Adaptation regularization: a general framework for transfer learning, IEEE Trans. Knowl. Data Eng. 26 (5) (2014) 1076–1089.
[5] L. Zhang, D. Zhang, Robust visual knowledge transfer via extreme learning machine-based domain adaptation, IEEE Trans. Image Process. 25 (10) (2016) 4959–4973.
[6] Y. Liu, L. Zhang, P. Deng, Z. He, Common subspace learning via cross-domain extreme learning machine, Cogn. Comput. 9 (4) (2017) 555–563.
[7] J. Hoffman, E. Rodner, J. Donahue, B. Kulis, K. Saenko, Asymmetric and category invariant feature transformations for domain adaptation, Int. J. Comput. Vis. 109 (1–2) (2014) 28–41.
[8] K. Saenko, B. Kulis, M. Fritz, T. Darrell, Adapting visual category models to new domains, in: Proceedings of the Computer Vision – ECCV 2010, 2010, pp. 213–226.
[9] B. Gong, Y. Shi, F. Sha, K. Grauman, Geodesic flow kernel for unsupervised domain adaptation, in: Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2012, pp. 2066–2073.
[10] J. Hoffman, E. Rodner, J. Donahue, T. Darrell, K. Saenko, Efficient learning of domain-invariant image representations, in: Proceedings of the International Conference on Learning Representations, 2013.
[11] B. Kulis, K. Saenko, T. Darrell, What you saw is not what you get: Domain adaptation using asymmetric kernel transforms, in: Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2011, pp. 1785–1792.
[12] B. Sun, J. Feng, K. Saenko, Return of frustratingly easy domain adaptation., in: Proceedings of the AAAI, 6, 2016, p. 8.
[13] B. Fernando, A. Habrard, M. Sebban, T. Tuytelaars, Unsupervised visual domain adaptation using subspace alignment, in: Proceedings of the 2013 IEEE International Conference on Computer Vision (ICCV), IEEE, 2013, pp. 2960–2967.
[14] S. Si, D. Tao, B. Geng, Bregman divergence-based regularization for transfer subspace learning, IEEE Trans. Knowl. Data Eng. 22 (7) (2010) 929–942.
[15] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, S. Saminger-Platz, Central moment discrepancy (CMD) for domain-invariant representation learning, International Conference on Learning Representations (2017).
[16] M. Gheisari, M.S. Baghshah, Unsupervised domain adaptation via representation learning and adaptive classifier learning, Neurocomputing 165 (2015) 300–311.
[17] N. Courty, R. Flamary, A. Habrard, A. Rakotomamonjy, Joint distribution optimal transportation for domain adaptation, in: Proceedings of the Advances in Neural Information Processing Systems, 2017, pp. 3733–3742.
[18] J. Huang, A. Gretton, K.M. Borgwardt, B. Schölkopf, A.J. Smola, Correcting sample selection bias by unlabeled data, in: Proceedings of the Advances in Neural Information Processing Systems, 2007, pp. 601–608.
[19] M. Chen, K.Q. Weinberger, J. Blitzer, Co-training for domain adaptation, in: Proceedings of the Advances in Neural Information Processing Systems, 2011, pp. 2456–2464.
[20] B. Tan, Y. Zhang, S.J. Pan, Q. Yang, Distant domain transfer learning., in: Proceedings of the AAAI, 2017, pp. 2604–2610.
[21] J. Yang, R. Yan, A.G. Hauptmann, Adapting svm classifiers to data with shifted distributions, in: Proceedings of the Seventh IEEE International Conference on Data Mining Workshops, 2007. ICDM Workshops 2007., IEEE, 2007, pp. 69–76.
[22] X. Li, W. Mao, W. Jiang, Extreme learning machine based transfer learning for data classification, Neurocomputing 174 (2016) 203–210.
[23] C. Chen, B. Jiang, X. Jin, Parameter transfer extreme learning machine based on projective model, in: Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), IEEE, 2018, pp. 1–8.
[24] M. Shao, D. Kit, Y. Fu, Generalized transfer subspace learning through low-rank constraint, Int. J. Comput. Vis. 109 (1–2) (2014) 74–93.
[25] L. Zhang, W. Zuo, D. Zhang, Lsdt: Latent sparse domain transfer learning for visual adaptation, IEEE Trans. Image Process. 25 (3) (2016) 1177–1191.
[26] L. Zhang, J. Yang, D. Zhang, Domain class consistency based transfer learning for image classification across domains, Inf. Sci. 418 (2017) 242–257.
[27] Y. Xu, X. Fang, J. Wu, X. Li, D. Zhang, Discriminative transfer subspace learning via low-rank and sparse representation, IEEE Trans. Image Process. 25 (2) (2016) 850–863.
[28] C. Chen, Z. Chen, B. Jiang, J. Jin, Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation, in: Thirty-Third AAAI Conference on Artificial Intelligence, 2019.
[29] J. Zhang, W. Li, P. Ogunbona, Joint geometrical and statistical alignment for visual domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[30] Y. Cao, M. Long, J. Wang, Unsupervised domain adaptation with distribution matching machines, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[31] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1) (2006) 489–501.
[32] G.-B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Trans. Syst. Man Cybern. Part B Cybern. 42 (2) (2012) 513–529.
[33] G. Huang, S. Song, J.N. Gupta, C. Wu, Semi-supervised and unsupervised extreme learning machines, IEEE Trans. Cybern. 44 (12) (2014) 2405–2417.
[34] W. Zong, G.-B. Huang, Y. Chen, Weighted extreme learning machine for imbalance learning, Neurocomputing 101 (2013) 229–242.
[35] L. Zhang, D. Zhang, Evolutionary cost-sensitive extreme learning machine, IEEE Trans. Neural Netw. Learn. Syst. 28 (12) (2017) 3045–3060.
[36] N.-Y. Liang, G.-B. Huang, P. Saratchandran, N. Sundararajan, A fast and accurate online sequential learning algorithm for feedforward networks, IEEE Trans. Neural Netw. 17 (6) (2006) 1411–1423.
[37] L.L.C. Kasun, Y. Yang, G.-B. Huang, Z. Zhang, Dimension reduction with extreme learning machine, IEEE Trans. Image Process. 25 (8) (2016) 3906–3918.
[38] J. Tang, C. Deng, G.-B. Huang, Extreme learning machine for multilayer perceptron, IEEE Trans. Neural Netw. Learn. Syst. 27 (4) (2016) 809–821.
[39] M. Uzair, A. Mian, Blind domain adaptation with augmented extreme learning machine features, IEEE Trans. Cybern. 47 (3) (2017) 651–660.
[40] L. Zhang, D. Zhang, Domain adaptation extreme learning machines for drift compensation in e-nose systems, IEEE Trans. Instrum. Measur. 64 (7) (2015) 1790–1801.
[41] L. Zhang, Z. He, Y. Liu, Deep object recognition across domains based on adaptive extreme learning machine, Neurocomputing 239 (2017) 194–203.
[42] S.M. Salaken, A. Khosravi, T. Nguyen, S. Nahavandi, Extreme learning machine based transfer learning algorithms: a survey, Neurocomputing 267 (2017) 516–524.
[43] B. Fernando, T. Tommasi, T. Tuytelaars, Joint cross-domain classification and subspace learning for unsupervised adaptation, Pattern Recogn. Lett. 65 (2015) 60–66.
[44] C. Ding, D. Zhou, X. He, H. Zha, R1-pca: rotational invariant ℓ_1-norm principal component analysis for robust subspace factorization, in: Proceedings of the Twenty-third International Conference on Machine Learning, ACM, 2006, pp. 281–288.
[45] F. Nie, H. Huang, X. Cai, C.H. Ding, Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization, in: Proceedings of the Advances in Neural Information Processing Systems, 2010, pp. 1813–1821.
[46] Q. Gu, Z. Li, J. Han, Joint feature selection and subspace learning, in: Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI, 22, 2011, p. 1294.
[47] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, Analysis of representations for domain adaptation, in: Proceedings of the Advances in Neural Information Processing Systems, 2007, pp. 137–144.
[48] M. Long, Y. Cao, J. Wang, M. Jordan, Learning transferable features with deep adaptation networks, in: Proceedings of the International Conference on Machine Learning, 2015, pp. 97–105.
[49] B. Sun, K. Saenko, Subspace distribution alignment for unsupervised domain adaptation., in: Proceedings of the BMVC, 2015, p. 24.
[50] J. Garcke, T. Vanck, Importance weighted inductive transfer learning for regression, in: Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2014, pp. 466–481.
[51] S.M. Salaken, A. Khosravi, T. Nguyen, S. Nahavandi, Seeded transfer learning for regression problems with deep learning, Expert Syst. Appl. 115 (2019) 565–577.
[52] G. Griffin, A. Holub, P. Perona, Caltech-256 object category dataset, California Institute of Technology, 2007.
[53] H. Bay, T. Tuytelaars, L. Van Gool, Surf: Speeded up robust features, in: Proceedings of the Computer Vision – ECCV 2006, 2006, pp. 404–417.
[54] M.-R. Amini, N. Usunier, C. Goutte, Learning from multiple partially observed views – an application to multilingual text categorization, in: Proceedings of the NIPS 22, 2009.
[55] Y.-H. Hubert Tsai, Y.-R. Yeh, Y.-C. Frank Wang, Learning cross-domain landmarks for heterogeneous domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5081–5090.

Chao Chen received the B.Eng. degree form Anhui University, China in 2016. He is currently a Ph.D. student at School of Information Science and Electrical Engineering, Zhejiang University, China. His research interests include Statistics and Optimization, Deep Learning and Transfer Learning. Also, he has published several papers in relevant forums, such as AAAI, IJCNN and Neural Processing Letters.

Boyuan Jiang received B.Eng. degree from Harbin Institute of Technology, China in 2017. He is now pursuing his M.Eng. in School of Information Science and Electrical Engineering, Zhejiang University, China. His research interests include Deep Learning, Computer Vision and Domain Adaptation. He has published several papers in relevant forums, such as Neural Computing and Applications and AAAI.

Zhaowei Cheng received B.Eng. degree from Chongqing University, China in 2018. she is currently working toward the Ph.D. degree at the School of Information Science and Electrical Engineering, Zhejiang University, China. Her research interests include machine learning, deep learning and domain adaptation.

Xinyu Jin is a professor in School of Information Science and Electrical Engineering, Zhejiang University, China. His research interests include Machine Learning, Deep Learning and Biomedical Image Processing. He is a member of the professional committee of the Chinese Institute of Electronics. Also, he has won several awards, such as Baosteel Outstanding Teacher Award, Second Prize in Science and Technology of Zhejiang Province in China and Second Prize of Zhejiang Teaching Achievements.