2014 IEEE Conference on Computer Vision and Pattern Recognition

Transfer Joint Matching for Unsupervised Domain Adaptation

Mingsheng Long[†‡], Jianmin Wang[†], Guiguang Ding[†], Jiaguang Sun[†], and Philip S. Yu[§]
[†]School of Software, TNList, Tsinghua University, Beijing, China
[‡]Department of Computer Science, Tsinghua University, Beijing, China
[§]Department of Computer Science, University of Illinois at Chicago, IL, USA
longmingsheng@gmail.com, {jimwang,dinggg,sunjg}@tsinghua.edu.cn, psyu@uic.edu

Abstract

Visual domain adaptation, which learns an accurate classifier for a new domain using labeled images from an old domain, has shown promising value in computer vision yet still been a challenging problem. Most prior works have explored two learning strategies independently for domain adaptation: feature matching and instance reweighting. In this paper, we show that both strategies are important and inevitable when the domain difference is substantially large. We therefore put forward a novel Transfer Joint Matching (TJM) approach to model them in a unified optimization problem. Specifically, TJM aims to reduce the domain difference by jointly matching the features and reweighting the instances across domains in a principled dimensionality reduction procedure, and construct new feature representation that is invariant to both the distribution difference and the irrelevant instances. Comprehensive experimental results verify that TJM can significantly outperform competitive methods for cross-domain image recognition problems.

1. Introduction

The exponential growth of online images and videos has created a compelling demand for automatic technologies for organizing and analyzing the multimedia content. Unfortunately, labeled images are usually very sparse in new visual domains. Moreover, it is very complex, if not impossible, to learn a visual category model without rich labeled images. In such real-world applications, it is indispensable in image classification to leverage abundant labeled images readily available in some old domains. Recently, the literature has witnessed increasing interests in developing *domain adaptation* [21] algorithms for cross-domain knowledge transfer problems. Domain adaptation has proven to be promising in image classification [25, 13] and tagging [23, 26], object recognition [15, 2, 7, 10], and feature learning [14, 12, 22]

In cross-domain problems, the source and target data are usually sampled from *different* probability distributions. Thus the major computational issue of domain adaptation is how to reduce the distribution difference between domains. Most recent works have explored two learning strategies independently for domain adaptation: (1) *feature matching*, which discovers a shared feature representation by jointly reducing the distribution difference and preserving the important properties of input data [19, 20, 16]; and (2) *instance reweighting*, which minimizes the distribution difference by reweighting the source data and then training a classifier on the reweighted source data [11, 3, 4, 5]. However, Figure 1 demonstrates a difficult setting: when the domain difference is substantially large, there will always exist some source instances that are not relevant to the target instances even in the feature-matching subspace. In this difficult setting, it is important and inevitable to perform joint feature matching and instance reweighting for robust domain adaptation. Recent works have also explored joint feature reweighting and subspace learning where irrelevant features are down-weighted [1, 17, 9]. However, in visual domain adaptation, the domain difference is substantially large and it is difficult to define the relevance of raw features to different domains.

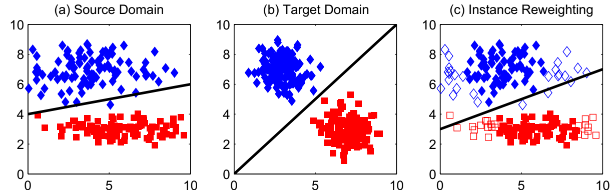


Figure 1. (a) source domain after feature matching; (b) target domain after feature matching. Due to the irrelevant source instances (shown as unfilled markers in c), the domain difference is still large after feature matching. (c) source domain after joint feature matching and instance reweighting. The irrelevant source instances are now down-weighted to further reduce domain difference (b vs c).

In this paper, we address the challenging setting in which the source and target domains are different in both feature distributions and instance relevances. We therefore propose a novel domain adaptation solution, referred to as *Transfer Joint Matching* (TJM), to jointly perform feature matching and instance reweighting across domains in a principled dimensionality reduction procedure. Specifically, we im-

1063-6919/14 $31.00 © 2014 IEEE
DOI 10.1109/CVPR.2014.183

plement feature matching by minimizing the nonparametric *Maximum Mean Discrepancy* (MMD) [8] in an infinite-dimensional reproducing kernel Hilbert space (RKHS), and implement instance reweighting by minimizing the $\ell_{2,1}$-norm structured sparsity penalty [17] on source instances. We integrate the minimization of MMD and $\ell_{2,1}$-norm with Principal Component Analysis (PCA) to construct domain-invariant feature representation that is effective for substantial domain difference. We present the learning algorithm with convergence analysis for TJM optimization problem.

We perform comprehensive experiments on 6 real-world datasets: digit (USPS+MNIST), object (MSRC+VOC2007 [16], Office+Caltech-256 [6]). From these datasets, we construct 16 *cross-domain* image datasets, each under different difficulty in knowledge adaptation. Our results demonstrate a significant improvement of **3.24%** in terms of the average classification accuracy, where TJM outperforms the state-of-the-art adaptation methods on most of datasets (10 out of 16). Our results reveal substantial effects of joint feature matching and instance reweighting for domain adaptation.

2. Related Work

According to the literature survey [21], existing domain adaptation methods can be roughly organized into two categories: *feature matching* and *instance reweighting*. Feature matching methods aim to reduce the distribution difference by learning a new feature representation. The feature representation can be learned via (1) extracting domain-invariant latent factors [12, 22, 6], (2) minimizing proper distance measures [19, 20, 16], and (3) reweighting relevant features with sparsity-promoting regularization [1, 17, 9]. Instance reweighting methods aim to reduce the distribution difference by reweighting the source instances according to their relevance to the target instances [11, 3, 4, 5]. However, all these methods have only explored feature matching and instance reweighting independently, and may not be effective enough when the domain difference is substantially large.

To our knowledge, our work is among the first attempts for visual domain adaptation which performs joint feature matching and instance reweighting in a principled dimensionality reduction procedure. The procedure is nontrivial, since we have to work in an infinite-dimensional RKHS to match the features more effectively, and no previous works have explored instance reweighting in such learning setting.

3. Transfer Joint Matching

In this section, we present the Transfer Joint Matching (TJM) approach for effective and robust domain adaptation.

3.1. Problem Definition

We begin with the definitions of terminologies. For clarity, the frequently used notations are summarized in Table 1.

Table 1. Notations and their descriptions used in this paper.

Notation	Description	Notation	Description
$\mathcal{D}_s, \mathcal{D}_t$	source/target domain	$\mathbf{X}$	input data matrix
n_s, n_t	#source/target examples	$\mathbf{K}$	input kernel matrix
m, C	#shared features/classes	$\mathbf{A}$	adaptation matrix
k	#subspace bases	$\mathbf{M}$	MMD matrix
λ	regularization parameter	$\mathbf{G}$	sub-gradient matrix

Notations: For a matrix $\mathbf{A} \in \mathbb{R}^{n\times k}$, denote the ith row as $\mathbf{a}^i$, the jth column as $\mathbf{a}_j$, the Frobenius norm as $\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^n \|\mathbf{a}^i\|_2^2}$, the $\ell_{2,1}$-norm as $\|\mathbf{A}\|_{2,1} = \sum_{i=1}^n \|\mathbf{a}^i\|_2$.

Definition 1 (**Domain**)**.** *A domain $\mathcal{D}$ is composed of an m-dimensional feature space $\mathcal{X}$ and a marginal probability distribution $P(\mathbf{x})$,* i.e.*, $\mathcal{D} = \{\mathcal{X}, P(\mathbf{x})\}$, where $\mathbf{x} \in \mathcal{X}$.*

Definition 2 (**Task**)**.** *Given domain $\mathcal{D}$, a task $\mathcal{T}$ is composed of a C-cardinality label set $\mathcal{Y}$ and a classifier $f(\mathbf{x})$,* i.e.*, $\mathcal{T} = \{\mathcal{Y}, f(\mathbf{x})\}$, where $y \in \mathcal{Y}$, and $f(\mathbf{x}) = Q(y|\mathbf{x})$ can be interpreted as the conditional probability distribution.*

Problem 1 (**Transfer Joint Matching**)**.** *Given a labeled source domain $\mathcal{D}_s = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n_s}, y_{n_s})\}$ and an unlabeled target domain $\mathcal{D}_t = \{\mathbf{x}_{n_s+1}, \ldots, \mathbf{x}_{n_s+n_t}\}$ under $\mathcal{X}_s = \mathcal{X}_t$, $\mathcal{Y}_s = \mathcal{Y}_t$, $P_s(\mathbf{x}_s) \neq P_t(\mathbf{x}_t)$, $Q_s(y_s|\mathbf{x}_s) \neq Q_t(y_t|\mathbf{x}_t)$, learn a new feature representation to reduce the domain difference by jointly (1) matching feature distributions, and (2) reweighting source instances across domains.*

3.2. Proposed Approach

In this paper, we propose to adapt different domains by a feature transformation T so that (1) the features are matched across domains through distance minimization, and (2) the source instances are reweighted through structured sparsity:

$$\min_{T\in\mathcal{H}} \left\|\mathbb{E}_{P(\mathbf{x}_s)}\left[T\left(\mathbf{x}_s\right)\right] - \mathbb{E}_{P(\mathbf{x}_t)}\left[T\left(\mathbf{x}_t\right)\right]\right\|^2 + \lambda\|T\|_{2,1} \quad (1)$$

There are two key factors that determine the approach: (1) the feature matching should be performed in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ to match both first- and high-order statistics; (2) the structured sparsity should be performed in the instance space instead of the feature space, otherwise we would have done feature reweighting rather than instance reweighting. These key factors motivate us to work in the RKHS, which is natural for both requirements.

3.2.1 Dimensionality Reduction

Dimensionality reduction methods can learn a transformed feature representation by minimizing the reconstruction error of the input data. For simplicity and generality, we will choose Principal Component Analysis (PCA) for data reconstruction. Denote $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{m\times n}$ the input data matrix, and $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}$ the centering matrix, where $n = n_s + n_t$ and $\mathbf{1}$ the $n \times n$ matrix of ones, then the covariance matrix can be computed as $\mathbf{XHX}^{\mathrm{T}}$. The learning goal of PCA is to find an orthogonal transformation matrix $\mathbf{V} \in \mathbb{R}^{m \times k}$ such that embedded data variance is maximized

$$\max_{\mathbf{V}^{\mathrm{T}}\mathbf{V}=\mathbf{I}} \operatorname{tr}\left(\mathbf{V}^{\mathrm{T}}\mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}\mathbf{V}\right) \tag{2}$$

where $\operatorname{tr}(\cdot)$ denotes the trace of a matrix. This optimization problem can be efficiently solved by eigendecomposition $\mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}\mathbf{V} = \mathbf{V}\mathbf{\Phi}$, where $\mathbf{\Phi} = \operatorname{diag}(\phi_1, \ldots, \phi_k) \in \mathbb{R}^{k \times k}$ are the k largest eigenvalues. Then we find the optimal k-dimensional representation by $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_n] = \mathbf{V}^{\mathrm{T}}\mathbf{X}$.

Kernelization: To work in the RKHS, consider kernel mapping $\psi : \mathbf{x} \mapsto \psi(\mathbf{x})$, or $\psi(\mathbf{X}) = [\psi(\mathbf{x}_1), \ldots, \psi(\mathbf{x}_n)]$, and kernel matrix $\mathbf{K} = \psi(\mathbf{X})^{\mathrm{T}}\psi(\mathbf{X}) \in \mathbb{R}^{n \times n}$. We utilize the Representer theorem $\mathbf{V} = \phi\left(\mathbf{X}\right)\mathbf{A}$ to kernelize PCA as

$$\max_{\mathbf{A}^{\mathrm{T}}\mathbf{A}=\mathbf{I}} \operatorname{tr}\left(\mathbf{A}^{\mathrm{T}}\mathbf{K}\mathbf{H}\mathbf{K}^{\mathrm{T}}\mathbf{A}\right) \tag{3}$$

where $\mathbf{A} \in \mathbb{R}^{n \times k}$ is the transformation matrix for Kernel-PCA, and the subspace embedding becomes $\mathbf{Z} = \mathbf{A}^{\mathrm{T}}\mathbf{K}$. Note that, through kernelization, now we can work in a possibly infinite-dimensional feature space, which can be easily manipulated in the instance space using the "kernel trick".

3.2.2 Feature Matching

However, even through the extracted k-dimensional representation, the distribution difference between the source and target domains will still be significantly large. Thus one major computational issue of domain adaptation is to reduce the difference in feature distributions by explicitly minimizing proper distance measures. Since parametrically estimating the probability density for a distribution is a nontrivial problem, we resort to match the first- and high-order statistics of different distributions. In this paper, we adopt the empirical *Maximum Mean Discrepancy* (MMD) [8, 19, 20] as the nonparametric distance measure to compare different distributions in the RKHS. MMD computes the distance between the empirical expectations of source and target data using k-dimensional embeddings extracted by Kernel-PCA:

$$\left\| \frac{1}{n_s} \sum_{i=1}^{n_s} \mathbf{A}^{\mathrm{T}}\mathbf{k}_i - \frac{1}{n_t} \sum_{j=n_s+1}^{n_s+n_t} \mathbf{A}^{\mathrm{T}}\mathbf{k}_j \right\|_{\mathcal{H}}^2 = \operatorname{tr}\left(\mathbf{A}^{\mathrm{T}}\mathbf{K}\mathbf{M}\mathbf{K}^{\mathrm{T}}\mathbf{A}\right) \tag{4}$$

where $\mathbf{M}$ is the MMD matrix and is computed as follows

$$M_{ij} = \begin{cases} \frac{1}{n_s n_s}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_s \\ \frac{1}{n_t n_t}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_t \\ \frac{-1}{n_s n_t}, & \text{otherwise} \end{cases} \tag{5}$$

By minimizing Equation (4) such that Equation (3) is maximized, the first- and high-order statistics of feature distributions are matched under the new representation $\mathbf{Z} = \mathbf{A}^{\mathrm{T}}\mathbf{K}$. Note that we just developed TJM to be similar to TCA [20].

3.2.3 Instance Reweighting

However, matching the feature distributions based on MMD minimization in Equation (4) is not good enough for domain adaptation, since it can only match the first- and high-order statistics, and the distribution matching is far from perfect. In particular, when the domain difference is substantially large, there will always exist some source instances that are not relevant to the target instances even in the TCA subspace. Therefore, an instance reweighting procedure should be cooperated with TCA to handle this difficult setting. Unfortunately, it is nontrivial to reweight source instances if we also require to match the feature distributions in the infinite-dimensional RKHS. Recent works have performed instance reweighting via kernel mean matching [11, 5], sample selection [28], and co-training [4]. But it remains unclear how to unify them with feature matching for better performance.

In this paper, we propose to impose the $\ell_{2,1}$-norm structured sparsity regularizer on the transformation matrix $\mathbf{A}$, which can introduce *row-sparsity* to the transformation matrix. Since each row of matrix $\mathbf{A}$ corresponds to an instance, the row-sparsity can essentially facilitate adaptive instance reweighting. We define the instance reweighting regularizer

$$\left\|\mathbf{A}_s\right\|_{2,1} + \left\|\mathbf{A}_t\right\|_F^2 \tag{6}$$

where $\mathbf{A}_s := \mathbf{A}_{1:n_s,:}$ is the transformation matrix corresponding to the source instances, and $\mathbf{A}_t := \mathbf{A}_{n_s+1:n_s+n_t,:}$ is the transformation matrix corresponding to the target instances. We only impose $\ell_{2,1}$-norm regularizer on source instances, since our aim is to reweight source instances by their relevance to the target instances. By minimizing Equation (6) such that Equation (3) is maximized, the source instances relevant (irrelevant) to the target instances are adaptively reweighted with greater (less) importance in the new representation $\mathbf{Z} = \mathbf{A}^{\mathrm{T}}\mathbf{K}$. With this regularizer, TJM is robust to the domain difference caused by irrelevant instances.

It is important to note that, our instance reweighting regularizer defined in Equation (6) is essentially different from the joint feature learning methods [1, 17, 9]. In joint feature learning, features are reweighted by their relevance to a specific domain. However, in visual domain adaptation problems, the domain difference is substantially large and it is difficult to define the relevance of features to different domains. Thus the joint feature learning methods may have to struggle for the cross-domain image recognition problems.

3.2.4 Optimization Problem

In this paper, we aim to reduce the domain difference by jointly matching the feature distributions and reweighting the source instances. By incorporating Equations (4) and (6) into Equation (3), we obtain the TJM optimization problem:

$$\min_{\mathbf{A}^{\mathrm{T}}\mathbf{K}\mathbf{H}\mathbf{K}^{\mathrm{T}}\mathbf{A}=\mathbf{I}} \operatorname{tr}\left(\mathbf{A}^{\mathrm{T}}\mathbf{K}\mathbf{M}\mathbf{K}^{\mathrm{T}}\mathbf{A}\right) + \lambda\left(\left\|\mathbf{A}_s\right\|_{2,1} + \left\|\mathbf{A}_t\right\|_F^2\right) \tag{7}$$

where λ is the regularization parameter to trade off feature matching and instance reweighting. To highlight its functionality, we term $\mathbf{A}$ as the *adaptation* matrix in the sequel. An important advantage of TJM is its capability to simultaneously match the feature distributions and reweight the source instances in a principled dimensionality reduction procedure. Thus TJM can be easy to implement and deploy.

3.3. Learning Algorithm

According to the constrained optimization theory, we denote $\mathbf{\Phi} = \mathrm{diag}(\phi_1, \ldots, \phi_k) \in \mathbb{R}^{k\times k}$ as the Lagrange multiplier, and derive the Lagrange function for problem (7) as

$$\begin{aligned} L = &\operatorname{tr}\left(\mathbf{A}^{\mathrm{T}}\mathbf{KMK}^{\mathrm{T}}\mathbf{A}\right) + \lambda\left(\|\mathbf{A}_s\|_{2,1} + \|\mathbf{A}_t\|_F^2\right) \\ &+ \operatorname{tr}\left(\left(\mathbf{I} - \mathbf{A}^{\mathrm{T}}\mathbf{KHK}^{\mathrm{T}}\mathbf{A}\right)\mathbf{\Phi}\right) \end{aligned} \tag{8}$$

Setting $\frac{\partial L}{\partial \mathbf{A}} = \mathbf{0}$, we obtain generalized eigendecomposition

$$\left(\mathbf{KMK}^{\mathrm{T}} + \lambda\mathbf{G}\right)\mathbf{A} = \mathbf{KHK}^{\mathrm{T}}\mathbf{A}\mathbf{\Phi} \tag{9}$$

$\|\mathbf{A}_s\|_{2,1}$ is a non-smooth function at zero, thus we compute its sub-gradient as $\frac{\partial\left(\|\mathbf{A}_s\|_{2,1} + \|\mathbf{A}_t\|_F^2\right)}{\partial\mathbf{A}} = 2\mathbf{GA}$, where $\mathbf{G}$ is a diagonal sub-gradient matrix with ith element equal to[1]

$$G_{ii} = \begin{cases} \frac{1}{2\|\mathbf{a}^i\|}, & \mathbf{x}_i \in \mathcal{D}_s, \mathbf{a}^i \neq \mathbf{0} \\ 0, & \mathbf{x}_i \in \mathcal{D}_s, \mathbf{a}^i = \mathbf{0} \\ 1, & \mathbf{x}_i \in \mathcal{D}_t \end{cases} \tag{10}$$

Finding the optimal adaptation matrix $\mathbf{A}$ is reduced to solving Equation (9) for the k smallest eigenvectors. Unfortunately, the sub-gradient matrix $\mathbf{G}$ is dependent on the adaptation matrix $\mathbf{A}$, which is also unknown beforehand. Thus, we resort to the alternating optimization strategy, where we iteratively update one variable with the other one fixed. The complete procedure is summarized in Algorithm 1. We will analyze the convergence of Algorithm 1 in Subsection 3.5.

3.4. Computational Complexity

Here we analyze the computational complexity of Algorithm 1 by the big O notation. Denote T the number of iterations, then typical values of k are not greater than 500, T not greater than 50, so $k \ll \min(m,n)$, $T \ll \min(m,n)$. The computational cost is detailed as follows: $O\left(mn^2\right)$ for computing the kernel matrix, *i.e.*, Line 2; $O\left(Tkn^2\right)$ for solving the generalized eigendecomposition problem with dense matrices, *i.e.*, Line 5; $O\left(Tn^2\right)$ for computing the sub-gradient matrix, *i.e.*, Line 6. In total, the computational complexity of Algorithm 1 is $O\left(Tkn^2 + mn^2\right)$. The complexity can be greatly reduced by low-rank approximation.

Algorithm 1: TJM: Transfer Joint Matching

```
  Input: Data $\mathbf{X}$; #subspace bases $k$, regularization parameter $\lambda$.
  Output: Adaptation matrix $\mathbf{A}$, embedding $\mathbf{Z}$, adaptive classifier $f$.
1 begin
2     Compute MMD matrix $\mathbf{M}$ by Equation (5), and kernel matrix
      $\mathbf{K}$ by $K_{ij} \leftarrow K(\mathbf{x}_i, \mathbf{x}_j)$ where $K(\cdot,\cdot)$ is a predefined kernel.
3     Set $\mathbf{M} \leftarrow \mathbf{M}/\|\mathbf{M}\|_F$, $\mathbf{G} \leftarrow \mathbf{I}$.
4     repeat
5         Solve the generalized eigendecomposition problem in
          Equation (9) and select the $k$ smallest eigenvectors to
          construct the adaptation matrix $\mathbf{A}$, and $\mathbf{Z} \leftarrow \mathbf{A}^{\mathrm{T}}\mathbf{K}$.
6         Update the sub-gradient matrix $\mathbf{G}$ by Equation (10).
7     until Convergence
8     Return an adaptive classifier $f$ trained on $\left\{\mathbf{A}^{\mathrm{T}}\mathbf{k}_i, y_i\right\}_{i=1}^{n_s}$.
```

3.5. Convergence Analysis

We prove that the alternating optimization procedure in Algorithm 1 converges to the optimal solution of $\mathbf{A}$ corresponding to optimization problem (7). Our proof is similar to methods in [18, 27]. We begin with the following lemma.

Lemma 1. *[18] The following inequality holds given that $\{\mathbf{v}_\tau^i\}_{i=1}^r$ are none-zero vectors, and r is arbitrary number.*

$$\sum_{i=1}^r \left(\left\|\mathbf{v}_{\tau+1}^i\right\| - \frac{\left\|\mathbf{v}_{\tau+1}^i\right\|^2}{2\left\|\mathbf{v}_\tau^i\right\|}\right) \leq \sum_{i=1}^r \left(\left\|\mathbf{v}_\tau^i\right\| - \frac{\left\|\mathbf{v}_\tau^i\right\|^2}{2\left\|\mathbf{v}_\tau^i\right\|}\right) \tag{11}$$

Theorem 1. *The iterative optimization in Algorithm 1,* i.e.*, Lines 4∼7, can monotonically decrease the objective function in Equation* (7) *in each iteration until convergence.*

Proof. Denote $\mathbf{A}_\tau$ and $\mathbf{G}_\tau$ the variables in iteration τ. Note that, $\mathbf{A}_\tau$ depends on $\mathbf{G}_{\tau-1}$ and $\mathbf{G}_\tau$ depends on $\mathbf{A}_\tau$. Based on the eigendecomposition principle, the computation of $\mathbf{A}_\tau$ in Line 5, Algorithm 1 satisfies following optimization

$$\mathbf{A}_\tau = \mathop{\arg\min}_{\mathbf{A}^{\mathrm{T}}\mathbf{KHK}^{\mathrm{T}}\mathbf{A}=\mathbf{I}} \operatorname{tr}\left(\mathbf{A}^{\mathrm{T}}\left(\mathbf{KMK}^{\mathrm{T}} + \lambda\mathbf{G}_{\tau-1}\right)\mathbf{A}\right)$$

For clarity, denote $\mathbf{W} = \mathbf{KMK}^{\mathrm{T}} + \lambda\mathbf{G}_t$, where $(\mathbf{G}_t)_{ii} = 1$ if $\mathbf{x}_t \in \mathcal{D}_t$ and $(\mathbf{G}_t)_{ii} = 0$ otherwise. Then we can derive

$$\begin{aligned} &\operatorname{tr}\left(\mathbf{A}_\tau^{\mathrm{T}}\left(\mathbf{W} + \lambda\mathbf{G}_{\tau-1}\right)\mathbf{A}_\tau\right) \leq \operatorname{tr}\left(\mathbf{A}_{\tau-1}^{\mathrm{T}}\left(\mathbf{W} + \lambda\mathbf{G}_{\tau-1}\right)\mathbf{A}_{\tau-1}\right) \\ &\Rightarrow \operatorname{tr}\left(\mathbf{A}_\tau^{\mathrm{T}}\mathbf{W}\mathbf{A}_\tau\right) + \lambda\sum_{i=1}^{n_s}\frac{\left\|\mathbf{a}_\tau^i\right\|^2}{2\left\|\mathbf{a}_{\tau-1}^i\right\|} \leq \operatorname{tr}\left(\mathbf{A}_{\tau-1}^{\mathrm{T}}\mathbf{W}\mathbf{A}_{\tau-1}\right) + \lambda\sum_{i=1}^{n_s}\frac{\left\|\mathbf{a}_{\tau-1}^i\right\|^2}{2\left\|\mathbf{a}_{\tau-1}^i\right\|} \\ &\Rightarrow \operatorname{tr}\left(\mathbf{A}_\tau^{\mathrm{T}}\mathbf{W}\mathbf{A}_\tau\right) + \lambda\sum_{i=1}^{n_s}\left\|\mathbf{a}_\tau^i\right\| - \lambda\sum_{i=1}^{n_s}\left(\left\|\mathbf{a}_\tau^i\right\| - \frac{\left\|\mathbf{a}_\tau^i\right\|^2}{2\left\|\mathbf{a}_{\tau-1}^i\right\|}\right) \\ &\leq \operatorname{tr}\left(\mathbf{A}_{\tau-1}^{\mathrm{T}}\mathbf{W}\mathbf{A}_{\tau-1}\right) + \lambda\sum_{i=1}^{n_s}\left\|\mathbf{a}_{\tau-1}^i\right\| - \lambda\sum_{i=1}^{m}\left(\left\|\mathbf{a}_{\tau-1}^i\right\| - \frac{\left\|\mathbf{a}_{\tau-1}^i\right\|^2}{2\left\|\mathbf{a}_{\tau-1}^i\right\|}\right) \end{aligned}$$

Using Lemma 1 and the definition of $\|\mathbf{A}\|_{2,1}$, we obtain

$$\operatorname{tr}\left(\mathbf{A}_\tau^{\mathrm{T}}\mathbf{W}\mathbf{A}_\tau\right) + \lambda\left\|\mathbf{A}_\tau^{(s)}\right\|_{2,1} \leq \operatorname{tr}\left(\mathbf{A}_{\tau-1}^{\mathrm{T}}\mathbf{W}\mathbf{A}_{\tau-1}\right) + \lambda\left\|\mathbf{A}_{\tau-1}^{(s)}\right\|_{2,1}$$

which establishes that the objective function in Equation (7) monotonically decreases under updates in Algorithm 1. □

[1] It is noteworthy that in practice, $\|\mathbf{a}^i\|$ can be very close to zero but not zero. However, $\|\mathbf{a}^i\|$ can be zero theoretically. In this case, we follow the regularization theory and define $G_{ii} = \frac{1}{2\|\mathbf{a}^i\|+\epsilon}$, where ϵ is a very small constant. It is easy to see that $\frac{1}{2\|\mathbf{a}^i\|+\epsilon}$ approximates $\frac{1}{2\|\mathbf{a}^i\|}$ when $\epsilon \to 0$.

4. Experiments

In this section, we conduct extensive experiments for image classification problems to evaluate the TJM approach. Datasets and codes will be available online on publication.

4.1. Data Preparation

USPS+MNIST, MSRC+VOC2007, and Office+Caltech-256 (see Figure 2 and Table 2) are six benchmark datasets widely adopted for visual domain adaptation algorithms.

USPS dataset consists of 7,291 training images and 2,007 test images of size 16×16.

MNIST dataset has a training set of 60,000 examples and a test set of 10,000 examples of size 28×28.

From Figure 2, we can see that USPS and MNIST follow very different distributions. They share 10 classes of digits. To speed up experiments, we construct one dataset *USPS vs MNIST* by randomly sampling 1,800 images in USPS to form the source data, and randomly sampling 2,000 images in MNIST to form the target data. We switch source/target pair to get another dataset *MNIST vs USPS*. We uniformly rescale all images to size 16×16, and represent each one by a feature vector encoding the gray-scale pixel values. Thus the source and target data can share the same feature space.

MSRC dataset is provided by Microsoft Research Cambridge, which contains 4,323 images labeled by 18 classes.

VOC2007 dataset (the training/validation subset) contains 5,011 images annotated with 20 concepts.

From Figure 2, we can see that MSRC and VOC2007 follow significantly different distributions, that is, MSRC is from standard images for benchmark evaluation, while VOC2007 is from arbitrary photos in Flickr. They share the following 6 semantic classes: “aeroplane”, “bicycle”, “bird”, “car”, “cow”, “sheep”. We follow [16] to construct one dataset *MSRC vs VOC* by selecting all 1,269 images in MSRC to form the source domain, and all 1,530 images in VOC2007 to form the target domain. Then we switch the source/target pair to get another dataset *VOC vs MSRC*. We uniformly rescale all images to be 256 pixels in length, and extract 128-dimensional dense SIFT (DSIFT) features using the VLFeat open source package. A 240-dimensional codebook is created, where Kmeans clustering is used to obtain the codewords. In this way, the training and test data are constructed to share the same label set and feature space.

Office [24, 6] is an increasingly popular benchmark for visual domain adaptation. The database contains three real-world object domains, **Amazon** (images downloaded from online merchants), **Webcam** (low-resolution images by a web camera), and **DSLR** (high-resolution images by a digital SLR camera). It has 4,652 images and 31 categories.

Caltech-256 is a standard database for object recognition. The database has 30,607 images and 256 categories.

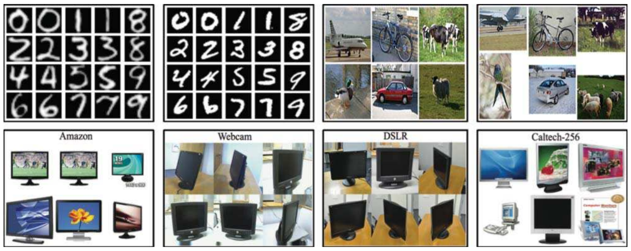


Figure 2. USPS, MNIST, MSRC, VOC2007, Office, Caltech-256.

Table 2. Statistics of the six benchmark digit and object datasets.

Dataset	Type	#Examples	#Features	#Classes	Subsets
USPS	Digit	1,800	256	10	USPS
MNIST	Digit	2,000	256	10	MNIST
MSRC	Object	1,269	240	20	MSRC
VOC2007	Object	1,530	240	68	VOC
Office	Object	1,410	800	10	A, W, D
Caltech-256	Object	1,123	800	10	C

In these expriments, we adopt the public Office+Caltech datasets released by Gong *et al*. [6]. SURF features are extracted and quantized into an 800-bin histogram with codebooks computed with Kmeans on a subset of images from *Amazon*. Then the histograms are standardized by z-score. Specifically, we have four domains, **C** (Caltech-256), **A** (Amazon), **W** (Webcam), and **D** (DSLR). By randomly selecting two different domains as source domain and target domain respectively, we construct $4 \times 3 = 12$ cross-domain object datasets, *e.g.*, $C \rightarrow A$, $C \rightarrow W$, $C \rightarrow D$, $\ldots$, $D \rightarrow W$.

4.2. Baseline Methods

We compare our TJM approach with five state-of-the-art (related) baseline methods for image recognition problems.

- 1-Nearest Neighbor Classifier (NN)
- Principal Component Analysis (PCA) + NN
- Joint Feature Selection and Subspace Learning (FSSL) [9] + NN
- Transfer Component Analysis (TCA) [20] + NN
- Geodesic Flow Kernel (GFK) [6] + NN

In particular, TCA is the most closely related method to TJM, while TJM differs from TCA by introducing an $\ell_{2,1}$-norm sparsity penalty on the source data, which can take advantage for joint feature matching and instance reweighting. As suggested by [6], NN is chosen as the base classifier since it does not require tuning cross-validation parameters.

4.3. Implementation Details

We follow the same evaluation protocol as [6, 20] for fair comparison. NN is trained on the labeled source data, and tested on the unlabeled target data; PCA, FSSL, TCA, GFK, and TJM are performed on all data as a dimensionality reduction procedure, then an NN classifier is trained on the labeled source data for classifying the unlabeled target data.

Under our experimental setup, it is impossible to tune the optimal parameters using cross validation, since labeled and unlabeled data are sampled from different distributions. Thus we evaluate all methods by empirically searching the parameter space for the optimal parameter settings which

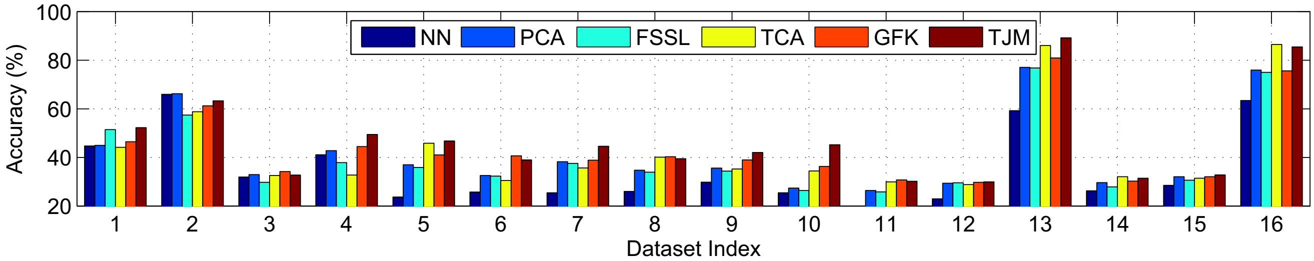


Figure 3. Recognition accuracy (%) on 16 cross-domain digit and object datasets, each under different difficulty in knowledge adaptation.

Table 3. Accuracy (%) on cross-domain digit and object datasets (bold and italic numbers indicate the best and second best results).

Dataset	Standard Learning		Transfer Learning			
	NN	PCA	FSSL	TCA	GFK	TJM
USPS vs MNIST (1)	44.7	44.95	*51.45*	44.15	46.45	**52.25**
MNIST vs USPS (2)	*65.94*	**66.22**	57.44	58.78	61.22	63.28
MSRC vs VOC (3)	31.96	*32.94*	29.74	32.55	**34.18**	32.75
VOC vs MSRC (4)	41.06	42.79	37.93	32.75	*44.47*	**49.41**
C→A (5)	23.70	36.95	35.88	*45.82*	41.02	**46.76**
C→W (6)	25.76	32.54	32.32	30.51	**40.68**	*38.98*
C→D (7)	25.48	38.22	37.53	35.67	*38.85*	**44.59**
A→C (8)	26.00	34.73	33.91	*40.07*	**40.25**	39.45
A→W (9)	29.83	35.59	34.35	35.25	*38.98*	**42.03**
A→D (10)	25.48	27.39	26.37	34.39	*36.31*	**45.22**
W→C (11)	19.86	26.36	25.85	29.92	**30.72**	*30.19*
W→A (12)	22.96	29.35	29.53	28.81	*29.75*	**29.96**
W→D (13)	59.24	77.07	76.79	*85.99*	80.89	**89.17**
D→C (14)	26.27	29.65	27.89	**32.06**	30.28	*31.43*
D→A (15)	28.50	32.05	30.61	31.42	*32.05*	**32.78**
D→W (16)	63.39	75.93	74.99	**86.44**	75.59	*85.42*
Average	35.01	41.42	40.16	42.79	43.86	**47.10**

gives the highest average accuracy on all datasets, and report the best results of each method. For subspace learning methods, we set #bases by searching $k \in [10, 20, \ldots, 200]$. For transfer learning methods, we set adaptation regularization parameter λ by searching $\lambda \in \{0.01, 0.1, 1, 10, 100\}$.

The TJM approach involves only two model parameters: #subspace bases k and regularization parameter λ. In the coming sections, we provide empirical analysis on parameter sensitivity, which verifies that TJM can achieve stable performance under a wide range of parameter values. When comparing with the baseline methods, we use a common set of parameter settings: $k = 20$ and $\lambda = 1.0$. The number of iterations for TJM to converge is $T = 10$. Similar to [20], we use Gaussian kernel with bandwidth set to the median squared distance between training instances. We do not run TJM repeatedly as it goes well with constant initialization.

We use classification *Accuracy* on test data as the evaluation metric, which is widely used in literature [20, 6, 16]

$$Accuracy = \frac{|\mathbf{x} : \mathbf{x} \in \mathcal{D}_t \wedge \widehat{y}(\mathbf{x}) = y(\mathbf{x})|}{|\mathbf{x} : \mathbf{x} \in \mathcal{D}_t|} \qquad (12)$$

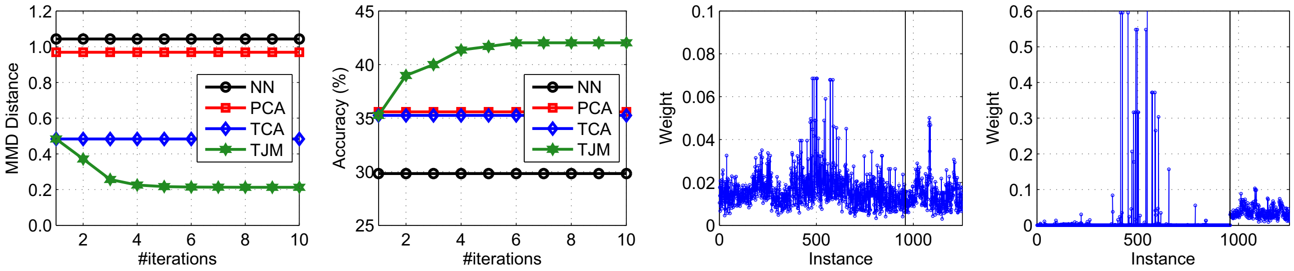
where $\mathcal{D}_t$ is the set of test data, $y(\mathbf{x})$ is the truth label of $\mathbf{x}$, $\widehat{y}(\mathbf{x})$ is the label predicted by the classification algorithm.

4.4. Experimental Results

The classification (recognition) accuracies of TJM and the five baseline methods on the 16 cross-domain digit and object datasets are illustrated in Table 3. For better interpretation, the results are also visualized in Figure 3. We observe that TJM achieves much better performance than the five baseline methods on most (10 out of 16) datasets. The average classification accuracy of TJM on the 16 datasets is **47.10%**, gaining a significant performance improvement of **3.24%** compared to the best baseline GFK. Note that, the adaptation difficulty in the 16 datasets varies a lot, since the standard NN classifier can only achieve an average classification accuracy of 35.01%, and may perform very poorly on many of the datasets. Although TJM cannot perform the best on all datasets, it is still established as an effective and robust approach due to the following aspects: (1) if it performs the best, then it usually outperforms the best baseline by a large margin, *e.g.*, on dataset $A \to D$; (2) otherwise, it performs only slightly worse than the best baseline. It verifies that TJM can construct more effective and robust representation for cross-domain image recognition problems.

Secondly, we notice that FSSL performs well on the digit datasets, but poorly on the object datasets. FSSL executes joint feature selection and subspace learning, in which a shared subspace is extracted while the relevant features are automatically selected for domain adaptation. However, in computer vision problems, the domain difference is substantially large, and thus it is nontrivial to define or select a set of relevant features that is invariant to different domains. Therefore, existing feature selection methods, *e.g.*, FSSL and Multi-Task Feature Learning (MTFL) [1], can perform well only when the relevant features can be adaptively selected. One such successful case is digit recognition, where the black-pixels of the image background can be defined as the irrelevant features and can be automatically filtered out. Another successful case is text classification, where it is natural to extract some semantically relevant words. Nevertheless, feature selection strategy is not as effective as instance reweighting for a wide range of computer vision problems.

Thirdly, TJM significantly outperforms TCA, which is a state-of-the-art domain adaptation method based on feature matching. TCA jointly executes feature transformation and feature matching in a reproducing kernel Hilbert space (RKHS), and thus is superior to PCA. However, as we have justified in this paper, only feature matching is not good



(a) MMD distance w.r.t. #iterations (b) Accuracy (%) w.r.t. #iterations (c) Instance weighting of TCA (d) Instance weighting of TJM

Figure 4. Effectiveness analysis for TJM: MMD distance, classification accuracy, and instance reweighting results on the $A \rightarrow W$ dataset.

enough for domain adaptation when the domain difference is substantially large, since there will always be some source instances which are not similar to the target instances even in the TCA-extracted subspace. TJM addresses TCA's limitation by introducing a structured sparsity penalty on the source instances, which can adaptively reweight the source instances according to their relevance to the target instances.

Lastly, we observe that GFK performs pretty well on the object datasets, but somewhat poorly on the digit datasets. In GFK, the subspace dimension should be small enough to ensure that different subspaces can transit smoothly along the geodesic flow. Then these infinite number of subspaces are implicitly extracted to encode the source and target data into a domain-invariant representation. However, the low-dimensional subspaces may not represent input data accurately when the input space is high-dimensional. TJM performs much better by learning an accurate shared subspace. It is important to note that, GFK outperforms TCA and underperforms TJM in general. This further verifies when the domain difference is substantially large, apart from feature matching, instance reweighting is also very important and inevitable for effective and robust visual domain adaptation.

4.5. Effectiveness Analysis

We further verify the effectiveness of TJM by inspecting the distribution distance and instance reweighting results.

Distribution Distance: We perform NN, PCA, TCA, and TJM on dataset $A \rightarrow W$ with the optimal parameter settings. Then we compute the MMD distance of each method on their induced embeddings by Equation (4). We note that, smaller distribution distance implies better generalization performance of the feature representation across domains.

Figure 4(a) shows the distribution distance computed for each method, and Figure 4(b) shows the classification accuracy. We can make these observations. (1) Without learning a feature representation, the distribution distance of NN in the original feature space is the largest. (2) PCA can learn a new representation in which the distribution distance is slightly reduced, thus it can help cross-domain problems only to some limited extent. (3) TCA can explicitly reduce the difference in the feature distributions, thus it can achieve better classification accuracy than PCA. (4) TJM can explicitly reduce the difference in both the feature space and instance space, thus it can extract the best representation in which the distribution distance is optimally minimized. By iteratively refining the source instance weighting through the structured sparsity penalty, TJM can increase (decrease) the weights of the relevant (irrelevant) source instances in each iteration to improve the classification performance.

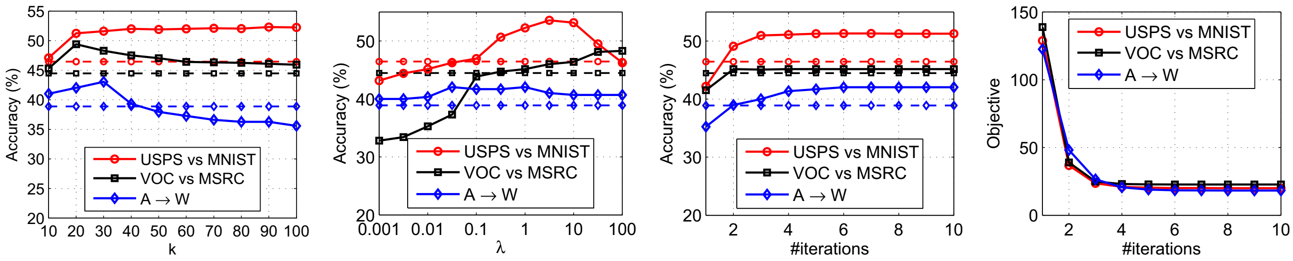
Instance Reweighting: We perform TCA and TJM on dataset $A \rightarrow W$ using their optimal parameter settings. Then we follow the definition of $\ell_{2,1}$-norm to compute the weighting of each instance $\mathbf{x}_i$ as $\|\mathbf{a}^i\|$ in the new feature subspace. Note that, in the adaptation matrix $\mathbf{A}$, each row corresponds to an instance, and each column corresponds to a subspace dimension. Thus the ℓ_2-norm of each row in $\mathbf{A}$, *i.e.*, $\|\mathbf{a}^i\|$, can essentially indicate the importance weighting of each instance in reconstructing the feature representation.

Figures 4(c) and 4(d) show the instance weighting results of TCA and TJM respectively, where a vertical line is plotted to separate source and target instances. An effective and robust representation for cross-domain problems requires that (1) the relevant source instances should be reweighted with greater importance, and (2) the irrelevant source instances should be reweighted with less importance. In this sense, we see that TCA weights the instances nearly equally, and thus cannot extract a good representation in which the source instances are reweighted by their relevance to the target instances.The significant performance improvement from TCA to TJM verifies that it is important and inevitable to jointly match features and reweight instances for effective and robust visual domain adaptation. It is great to see that TJM can learn the ideal instance weighting, *i.e.*, irrelevant source instances are adaptively filtered out to reduce the domain difference. Thus TJM can extract a domain-invariant representation for much better generalization performance.

4.6. Parameter Sensitivity

We conduct sensitivity analysis on the *USPS vs MNIST*, *VOC vs MSRC*, and $A \rightarrow W$ datasets, while similar trends on all other datasets are not shown due to space limitation.

We run TJM with varying values of k. It can be chosen



(a) #subspace bases k (b) regularization parameter λ (c) accuracy w.r.t. #iterations (d) objective w.r.t. #iterations

Figure 5. Parameter sensitivity and convergence study for TJM on digit and object datasets (dashed lines show the best baseline results).

Table 4. Time complexity of TJM and all the baseline methods.

Method	Runtime (s)	Method	Runtime (s)	Method	Runtime (s)
NN	0.31	PCA	0.42	FSSL	3.45
TCA	0.87	GFK	1.29	TJM	6.88

such that the low-dimensional representation is accurate for data reconstruction. We plot classification accuracy w.r.t. different values of k in Figure 5(a), and choose $k \in [10, 50]$.

We run TJM with varying values of λ. Intuitively, when $\lambda \to 0$, TJM optimization problem degenerates. When $\lambda \to \infty$, the joint feature matching and instance reweighting is not performed. We plot classification accuracy w.r.t. different values of λ in Figure 5(b), and choose $\lambda \in [0.1, 10.0]$.

4.7. Convergence and Time Complexity

We empirically check the convergence property of TJM. Figures 5(c) and 5(d) show that classification accuracy (objective function) increases (decreases) steadily with more iterations and converges within only 10 iterations.

We check time complexity on dataset $A \to W$ with 800 features and 1,253 images, and show the results in Table 4. We see that TJM iterates T-times and is worse than TCA.

5. Conclusion and Future Work

In this paper, we have proposed a novel Transfer Joint Matching (TJM) approach for visual domain adaptation problems. TJM aims to jointly match features and reweight instances across domains in a principled dimensionality reduction procedure. An important advantage of TJM is that it is robust to both the distribution difference and the irrelevant instances. Comprehensive experimental results show that TJM is effective for a variety of cross-domain problems, and can significantly outperform state-of-the-art adaptation methods even if the domain difference is substantially large.

In the future, we plan to extend joint matching strategy to alternative sparse learning methods, *e.g.*, Sparse Coding.

References

[1] A. Argyriou and T. Evgeniou. Multi-task feature learning. In *NIPS*, 2006.

[2] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *ICCV*, 2011.

[3] L. Bruzzone and M. Marconcini. Domain adaptation problems: A dasvm classification technique and a circular validation strategy. *TPAMI*, 2010.

[4] M. Chen, K. Q. Weinberger, and J. C. Blitzer. Co-training for domain adaptation. In *NIPS*, 2011.

[5] W.-S. Chu, F. De la Torre, and J. F. Cohn. Selective transfer machine for personalized facial action unit detection. In *CVPR*, 2013.

[6] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012.

[7] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, 2011.

[8] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Scholkopf, and A. J. Smola. A kernel method for the two-sample problem. In *NIPS*, 2006.

[9] Q. Gu, Z. Li, and J. Han. Joint feature selection and subspace learning. In *IJCAI*, 2011.

[10] M. Guillaumin and V. Ferrari. Large-scale knowledge transfer for object localization in imagenet. In *CVPR*, 2012.

[11] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Scholkopf. Correcting sample selection bias by unlabeled data. In *NIPS*, 2006.

[12] I.-H. Jhuo, D. Liu, D.-T. Lee, and S.-F. Chang. Robust visual domain adaptation with low-rank reconstruction. In *CVPR*, 2012.

[13] L. Jie, T. Tommasi, and B. Caputo. Multiclass transfer learning from unconstrained priors. In *ICCV*, 2011.

[14] C. H. Lampert and O. Krömer. Weakly-paired maximum covariance analysis for multimodal dimensionality reduction and transfer learning. In *ECCV*, 2010.

[15] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009.

[16] M. Long, G. Ding, J. Wang, J. Sun, Y. Guo, and P. S. Yu. Transfer sparse coding for robust image representation. In *CVPR*, 2013.

[17] M. Masaeli, G. Fung, and J. G. Dy. From transformation-based dimensionality reduction to feature selection. In *ICML*, 2010.

[18] F. Nie, H. Huang, X. Cai, and C. Ding. Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization. In *NIPS*, 2010.

[19] S. J. Pan, J. T. Kwok, and Q. Yang. Transfer learning via dimensionality reduction. In *AAAI*, 2008.

[20] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE TNN*, 22(2):199–210, 2011.

[21] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE TKDE*, 22:1345–1359, 2010.

[22] Q. Qiu, V. M. Patel, P. Turaga, and R. Chellappa. Domain adaptive dictionary learning. In *ECCV*, 2012.

[23] S. D. Roy, T. Mei, W. Zeng, and S. Li. Socialtransfer: Cross-domain transfer learning from social streams for media applications. In *ACM MM*, 2012.

[24] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *ECCV*, 2010.

[25] H. Wang, F. Nie, H. Huang, and C. Ding. Dyadic transfer learning for cross-domain image classification. In *ICCV*, 2011.

[26] S. Wang, S. Jiang, Q. Huang, and Q. Tian. Multi-feature metric learning with knowledge transfer among semantics and social tagging. In *CVPR*, 2012.

[27] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou. $\ell_{2,1}$-norm regularized discriminative feature selection for unsupervised learning. In *IJCAI*, 2011.

[28] E. Zhong, W. Fan, J. Peng, K. Zhang, J. Ren, D. Turaga, and O. Verscheure. Cross domain distribution adaptation via kernel mapping. In *KDD*, 2009.