CROSS-DOMAIN RECOGNITION BY IDENTIFYING COMPACT JOINT SUBSPACES

Yuewei Lin[a], *Jing Chen*[b], *Yu Cao*[c], *Youjie Zhou*[a], *Lingfeng Zhang*[d], *Song Wang*[a]

a,University of South Carolina b,University of Macau c,IBM Research - Almaden d,University of Houston

ABSTRACT

This paper introduces a new method to solve the cross-domain recognition problem. Different from the traditional domain adaption methods which rely on a global domain shift for all classes between source and target domain, the proposed method is more flexible to capture individual class variations across domains. We propose to solves the problem by finding the compact joint subspaces of source and target domain. We evaluate the proposed method on two widely used datasets and comparison results demonstrates that the proposed method outperforms the comparison methods.

1. INTRODUCTION

Traditional machine learning methods often assume that the training data and testing data are from same feature space and following similar distributions. However this assumption may not be true in many real applications. Namely the training data is obtained from one domain, while the testing data is coming from a different domain. For example, Figure 1 shows coffee-mug images collected from four different domains (Amazon, Caltech256, DSLR and Webcam), which present different image resolutions, viewpoints, background complexities and object layout patterns, etc. These domain differences lead to a dilemma that 1) labeling data in each domain for training would be expensive; and 2) directly applying the classifiers from one domain to another may result in significant degraded performance [1]. The dilemma consequently poses the cross-domain recognition problem, namely how to utilize the labeled data in a *source* domain to classify/recognize the unlabeled data in a *target* domain.

To achieve cross-domain recognition, a number of *Domain Adaptation* (DA) methods have been developed to adapt the classifier for one domain to another [2]. The existing DA algorithms can be either in a (semi-)supervised domain way or a unsupervised way, based on the availability of labeled data from the target domain. (Semi-)supervised DA assumes that there are some labeled data available in the target domain [3, 4, 5, 6]. Recently, the subspace based DA has been found to be effective to handle cross-domain problem [6, 7, 8, 9, 10]. They either constructed a set of intermediate subspaces for modeling the shifts between domains [7, 8, 10], or generated a domain-invariant subspace in which the data from source and target domains can represent each other well [9, 6, 11].

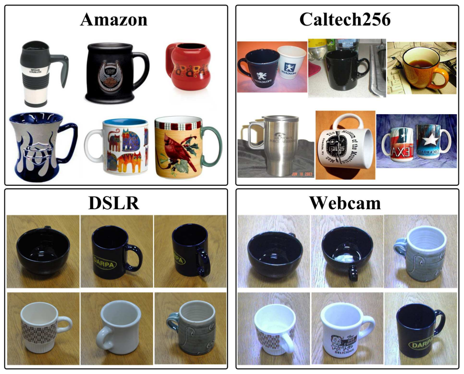

Fig. 1. Sample images from four different domains.

All these methods mentioned above utilize the data from each domain all together to generate a single subspace for each domain. In practice, however, the intrinsic feature shift of each class may not be exactly the same. The existing methods can obtain a global domain shift, but ignore the individual class difference across domains.

To circumvent the limitation of the global domain shift, we adopt a natural and widely used assumption that "the data samples from the same class should lay on a low-dimensional subspace, even if they come from different domains [12]". This assumption not only holds on many computer vision tasks, e.g. [13], but also is used as a human cognitive mechanism for visual object recognition [14]. Note that this assumption doesn't mean that the target data samples exactly lay on the subspace of the source data points, since different domains show subspace shift [10]. Figure 2 gives an illustration of a compact joint subspace covering source and target domains for a specific class. The source and target subspaces have the overlap which implicitly represents the intrinsic characteristics of the considered class. They have their own exclusive bases because of the domain shift, such as the varying illumination or changing the view perspectives. Based on the above assumption, we propose a new method that solves the cross-domain recognition by finding the compact joint subspaces of source and target domain. Specifically, we construct subspaces for each of the classes in labeled source domain. Then we construct subspaces in the target domain, called *anchor subspaces*, by collecting unlabeled samples that are close to each other and highly likely all fall into the same class. The corresponding class label is then assigned by minimizing a cost function. We further construct the compact joint subspaces for each class by combining the anchor subspace to corresponding source subspaces. Finally, the SVM classifier is trained using the samples in the compact joint subspace.

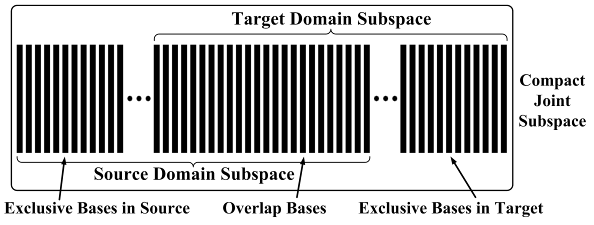

Fig. 2. An illustration of a compact joint subspace between source and target domains for a specific class. This subspace consists of overlap bases between domains, which represent the intrinsic characteristics of this class implicitly, and exclusive bases of different domains, which represent their own exclusive characteristics.

The contributions of this paper are: **1)** by assuming that the data samples from one specific class, even though they come from different domains, should lay on a low dimensional subspace, we generate one compact joint subspace for each class independently. Each compact joint subspace carries the information not only about the intrinsic characteristics of the corresponding class, but also about the specificity for each domain. **2)** To construct the compact joint subspaces, we first generate anchor subspaces in the target domain, assign labels to them, and combine these anchor subspaces to the corresponding source subspaces. **3)** We propose a cost function that implicitly maximizes the overlap between source subspace and target subspace for each class as well as maintains the topological structure in the target domain.

Note that we use the data samples themselves as the over-complete bases to represent the subspace implicitly instead of getting the orthogonal bases for the subspaces.

2. PROPOSED MODEL

Suppose two sets of data samples, $\{X_i^S\}_{i=1}^{N_S} \in R^{d\times N_S}$ and $\{X_i^T\}_{i=1}^{N_T} \in R^{d\times N_T}$ are from source domain $\mathcal{S}$ and target domain $\mathcal{T}$, respectively, where d is the data dimension. The labels of data samples in $\mathcal{S}$, denoted as $Y^S = \{y_i^S\}_{i=1}^{N_S} \in R^{C\times N_S}$, are known, where C is the number of classes, $y_i^S \in \{0,1\}^C$ is a C bit binary code of the ith data sample in source domain. If this data sample belongs to class j, the jth bit of y_i^S is 1 and all other bits are 0. Our aim is to estimate the labels of data samples in the target domain $Y^T \in R^{C\times N_T}$.

The proposed algorithm consists five steps: **(1)** We construct a set of subspaces $M_i^{\mathcal{S}} = \{X_j^S|_{X_j^S \in C_i}\}$, one for each class in the source domain, illustrated in Figure. 3(a). For each class, we simply take all the data that belong to this subspace as its over-complete bases. **(2)** We construct a number of anchor subspaces in the target domain, denoted as

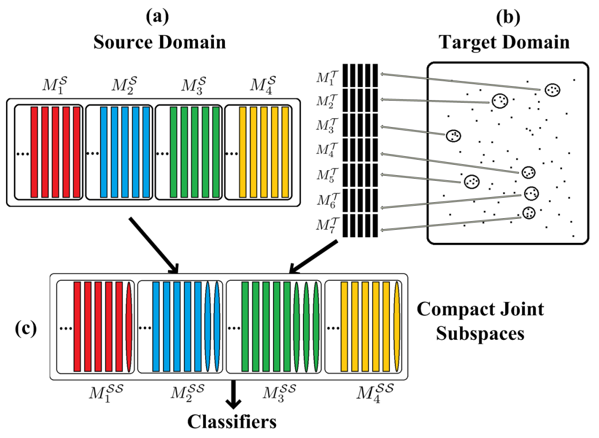

Fig. 3. The overview of the proposed model. (a). Anchor subspaces construction. Data samples in each circle denote a core subgroup and they construct an anchor subspace as one row of black bars. (b). Subspaces for each class in source domain. The bars with the same color denote the bases of one class. (c). Compact joint subspaces construction. The ellipses denote the bases from anchor subspaces.

$\{M_i^{\mathcal{T}}\}_{i=1}^K$, by grouping target data samples with high similarities. **(3)** We assign a label for each anchor subspace by minimizing a cost function which reflects a) the cross-domain distance between the anchor subspace and the corresponding source subspace; and b) the within-domain topological relation of the anchor subspaces in the target domain. **(4)** We construct compact joint subspace $M_i^{SS} = \{M_i^{\mathcal{S}}, M_j^{\mathcal{T}}|_{M_j^{\mathcal{T}} \in C_i}\}$, where C_i denotes the ith class, as illustrated in Figure. 3(c). **(5)** We train one-vs-rest SVM classifiers for each class using the labeled data in the compact joint subspace.

2.1. Anchor subspaces obtained in target domain

We construct each anchor subspace by selecting one target data sample and combining its nearest neighbors. This way, the obtained compact group of data samples are likely to be from the same class [15]. Specifically, we first apply the K-means algorithm to cluster all the target data into a large number of Z groups. We set $Z = \frac{N_T}{\gamma}$, where γ is the desired average group size. In each group of data, we find a compact *core subgroup* consisting of a small number of N samples, which are taken for constructing an anchor subspace. We construct the core subgroup for the group L is constructed as follow. First, we estimate the center of the core subgroup by finding the data sample x^* to $\min_{x\in L} \sum_{y\in\mathcal{N}_{(N-1)}(x)} \|x-y\|_2$, where $\mathcal{N}_{(N-1)}(x)$ denotes the $N-1$ nearest neighbors of x in L. Then, Take $x^* \cup \mathcal{N}_{(N-1)}(x^*)$ as the core subgroup for constructing an anchor subspace.

2.2. Labeling each anchor subspace

We have constructed C subspaces in the source domain, $\{M_i^S\}_{i=1}^C$ and their corresponding labels, $Y = \{y_i\}_{i=1}^C \in$

$R^{C\times C}$, in which the ith bit of y_i is 1 and all the other bits of y_i are 0. In this section, we aim to assign class labels $Y' = \{y'_i\}_{i=1}^{K} \in R^{C\times K}$ for the K anchor subspaces $\{M_i^{\mathcal{T}}\}_{i=1}^{K}$ constructed in target domain.

2.2.1. *Distance between subspaces.*

To calculate the distance between two subspaces, principal angles are usually used [8, 16]. In this paper, followed by [16], we define distance between two subspaces, e.g. M_i with p data samples and M_j with q data samples as follow: we first orthogonalize both of them to obtain $\mathcal{M}_i$ and $\mathcal{M}_j$, and then calculate the distance as: $D(M_i, M_j) \triangleq \sum_{m=1}^{min(p,q)} \sin\theta_m$, where θ_m come from the SVD of $(\mathcal{M}_i)'\,\mathcal{M}_j$, that $(\mathcal{M}_i)'\,\mathcal{M}_j = U(\cos\Theta)V'$. We then generate two affinity matrices, A^{ST} reflect the distances between anchor subspaces and source subspaces, A^{TT} reflects the pairwise distances among anchor subspaces, i.e., $A^{ST}_{ij} = \exp\left(-\frac{D(M_i^{\mathcal{S}}, M_j^{\mathcal{T}})}{2\sigma^2}\right)$ and $A^{TT}_{ij} = \exp\left(-\frac{D(M_i^{\mathcal{T}}, M_j^{\mathcal{T}})}{2\sigma^2}\right)$, where $M_i^{\mathcal{S}}$ and $M_j^{\mathcal{T}}$ denote the subspaces from the source and target domains, respectively.

2.2.2. *Cost function and optimization*

Two important issues are considered in assigning a label to each anchor subspace: 1) the distance between an anchor subspace and the same-label source subspace should be small, and 2) the local topological structures in the target domain should be preserved [17], i.e., anchor subspaces with shorter distance are more preferable to be assigned to the same class. We propose the cost function as follow:

$$\mathcal{C}(Y') = \sum_{i=1}^{C}\sum_{j=1}^{K} \|y_i - y'_j\|^2 A^{ST}_{ij} + \rho \sum_{j=1}^{K}\sum_{j'=1}^{K} \|y'_i - y'_j\|^2 A^{TT}_{jj'} \tag{1}$$

Adding a constant term $\sum_{i=1}^{C}\sum_{j=1}^{C} \|y_i - y_j\|^2 I_{ij}$ into $\mathcal{C}$ and splitting the first term into two parts, the cost function $\mathcal{C}$ can be written as:

$$\mathcal{C}(Y') = \sum_{i=1}^{C+K}\sum_{j=1}^{C+K} \|\mathcal{Y}_i - \mathcal{Y}_j\|^2 \mathcal{A}_{ij},\ s.t.\ \mathcal{Y}^{\mathrm{T}}\mathbf{1} = \mathbf{1}. \tag{2}$$

where $\mathcal{Y} = [Y, Y']$, $\mathcal{A} = \begin{bmatrix} I & \frac{1}{2}A^{ST} \\ \frac{1}{2}\left(A^{ST}\right)^{\mathrm{T}} & \rho A^{TT} \end{bmatrix}$. We also relax the constraint to this cost function by only requiring the sum of each row in $\mathcal{Y}$ to be 1.

By including the constraint term, the cost function can be written in a matrix form [18]

$$\mathcal{L}(\mathcal{Y}, \lambda) = Tr\left(\mathcal{Y}\Delta\mathcal{Y}^{\mathrm{T}}\right) + \lambda^{\mathrm{T}}\left(\mathcal{Y}^{\mathrm{T}}\mathbf{1} - \mathbf{1}\right) + \frac{\mu}{2}\|\mathcal{Y}^{\mathrm{T}}\mathbf{1} - \mathbf{1}\|_2^2, \tag{3}$$

where $\Delta = \mathcal{D} - \mathcal{C}$ is the *Laplacian matrix* of $\mathcal{A}$. $\lambda \in \mathcal{R}^{C+K}$ is the Lagrange multiplier. To minimize the objective function $\mathcal{L}$, we alternately update its two unknowns $\mathcal{Y}$ and λ:

(1) Having λ fixed, optimize $\mathcal{Y}$ by computing the derivative of $\mathcal{L}$ with the respect to $\mathcal{Y}$ and setting it to be zero:

$$\frac{\partial\mathcal{L}(\mathcal{Y}, \lambda)}{\partial\mathcal{Y}} = 0 \Rightarrow \mathcal{Y}\Delta + \mathbf{1}\lambda^{\mathrm{T}} + \mu\left(\mathbf{1}\mathbf{1}^{\mathrm{T}}\mathcal{Y} - \mathbf{1}\mathbf{1}^{\mathrm{T}}\right) = 0. \tag{4}$$

We first split the Laplacian matrix Δ into 4 blocks along the Cth row and column as in [19] $\Delta = \begin{bmatrix} \Delta_{CC} & \Delta_{CK} \\ \Delta_{KC} & \Delta_{KK} \end{bmatrix}$, and split λ to: $\lambda_I = [\lambda_1, \cdots, \lambda_C]^{\mathrm{T}}$ and $\lambda_{II} = [\lambda_{C+1}, \cdots, \lambda_{C+K}]^{\mathrm{T}}$. Then Y' can be updated by solving the following equation:

$$Y'^{(k+1)}\Delta_{KK} + \mu\mathbf{1}\mathbf{1}^{\mathrm{T}}Y'^{(k+1)} = \mu\mathbf{1}\mathbf{1}^{\mathrm{T}} - Y\Delta_{CK} - \mathbf{1}\lambda_{II}^{(k)\mathrm{T}}. \tag{5}$$

With this solution, $\mathcal{Y}^{(k+1)}$ can be achieved by putting Y and $Y'^{(k+1)}$ together as $\mathcal{Y}^{(k+1)} = [Y, Y'^{(k+1)}]$.

(2) Having $\mathcal{Y}$ fixed, perform a gradient ascending update with the step of μ on Lagrange multipliers as:

$$\lambda^{(k+1)} = \lambda^{(k)} + \mu\left({\mathcal{Y}^{(k+1)}}^{\mathrm{T}}\mathbf{1} - \mathbf{1}\right). \tag{6}$$

We initialize $\lambda^{(0)}$ and $Y'^{(0)}$ to be zero, and set the maximal number of iteration $maxIter$ to be 10000. Finally, we set the bit with the maximal value in each row to 1 and all the other bits to 0 after we get Y'.

3. EXPERIMENTAL RESULTS

In the section, we evaluate the proposed algorithm on two widely used cross domain recognition datasets: object recognition image dataset and sentiment classification dataset.

3.1. Cross-domain dataset for object recognition

The first dataset we evaluate on is a object recognition image dataset, which including has four sub-datasets, i.e., Amazon, DSLR Webcam and Caltech [1], which we use as four domains. There are 2533 images from 10 classes in total. Following the same way of feature extraction in previous works (e.g.[10]), we use SURF [20] descriptor to extract features for each image. We ran our algorithm 20 times for each task and give the average accuracy rate (%) and standard deviation (%).

First, we report the results on single source- and target-domain settings. It can be seen in Table. 1 that our algorithm performs best in 6 out 8 domain pairs. We only show 8 pairs out of 12 in total due to the page limitation, and we actually reach the best performance in 9 out of 12 domain pairs. Note that the "Metric" method [4] is a semi-supervised method.

Then we evaluate the performance when there are multiple source or target domains. When there are multiple source domains, as shown in Table. 2, it is clearly to see that the proposed method outperforms all the comparison methods significantly. For SGF [7], we report its performance under both

[1]For simplicity, hereafter we use "A", "C", "D" and "W" to denote the "Amazon", "Caltech", "DSLR" and "Webcam" domains, respectively.

Table 1. Results of single source and target domain on the object recognition dataset. "-" denotes that there is no resualt reported before.

Model	C→A	C→D	A→C	A→W	D→A	D→W	W→A	W→C
SGF [7]	36.8±0.5	32.6±0.7	35.3±0.5	31.0±0.7	32.0±0.4	66.0±0.5	27.5±0.5	21.7±0.4
GFK [8]	40.4±0.7	41.1±1.3	37.9±0.4	35.7±0.9	36.1±0.4	79.1±0.7	35.5±0.7	29.3±0.4
Metric [4]	33.7±0.8	35.0±1.1	36.0±1.0	21.7±0.5	30.3±0.8	55.6±0.7	38.6±0.8	32.3±0.8
ITL [21]	49.2±0.6	44.4±1.2	38.5±0.4	40.0±1.3	39.6±0.4	83.6±0.5	-	-
SI [10]	45.4±0.3	42.3±0.4	40.4±0.5	37.9±0.9	39.1±0.5	86.2±1.0	38.3±0.3	**36.3±0.3**
SA(SVM) [11]	46.1	39.4	39.9	39.6	**42.0**	82.3	39.3	31.8
CJS (ours)	**59.1±1.2**	**53.0±3.5**	**47.6±1.1**	**42.2±2.9**	37.9±1.6	**89.3±1.7**	**39.5±1.3**	33.5±1.6

Table 2. The results of multi-source domain adaptation on the object recognition dataset. Note that all the comparison methods are semi-supervised domain adaptation ones.

Model	D, A→W	A, W→D	W, D→A
SGF [7] (US)	31.0±1.6	25.0±0.4	15.0±0.4
SGF [7] (SS)	52.0±2.5	39.0±1.1	28.0±0.8
RDALR [9]	36.9±1.1	31.2±1.3	20.9±0.9
FDDL [22]	41.0±2.4	38.4±3.4	19.0±1.2
SDDL [6]	57.8±2.4	56.7±2.3	24.1±1.6
CJS (ours)	**73.2±2.5**	**81.3±1.3**	**41.1±1.1**

unsupervised (US) and semi-supervised (SS) settings. When there are multiple target domains, we only find one comparison method, SGF [7]. It can be seen that the proposed algorithm performs better than both of unsupervised (US) and semi-supervised (SS) settings of SGF.

Table 3. The results of multi-target domain adaptation on the object recognition dataset.

Model	W→A,D	D→A,W	A→D,W
SGF [7] (US)	28.0±1.9	35.0±1.7	22.0±0.2
SGF [7] (SS)	42.0±2.8	46.0±2.3	32.0±0.9
CJS (ours)	**45.1±1.2**	**48.4±2.2**	**44.2±2.0**

It is worth to mention that, based on our experiment, the proposed algorithm is not sensitive to the following two main parameters: the desired average size of each group constructed by using K-means algorithm, γ, and the number of data samples in each anchor subspace, N. In our experiments, we consistently set γ to be 20 and N to be 5.

3.2. Cross-domain dataset for sentiment classification

We also evaluate the proposed algorithm in a domain adaptation task from the natural language processing area. In this task, customers' reviews on four different products (kitchen applications, DVDs, books and electronics) are collected as four domains [23]. The goal of this task is adapt the classifier training on one domain and use it for classifying data samples in another domain. It is clear to see, in Table 4, that overall the proposed algorithm outperforms other 7 methods.

Table 4. Domain adaptation results on the sentiment classification. K: kitchen, D: dvd, B: books, E: electronics

Model	K→D	D→B	B→E	E→K
TCA [24]	60.4	61.4	61.3	68.7
SGF [7]	67.9	68.6	66.9	75.1
FGK [8]	69.0	71.3	68.4	78.2
SCL [25]	72.8	76.2	75.0	82.9
KMM [26]	72.2	78.6	76.9	83.5
Metric [4]	70.6	72.0	72.2	77.1
Landmark [27]	75.1	**79.0**	78.5	83.4
CJS (ours)	**77.8**	77.0	**83.2**	**84.1**

4. CONCLUSION

This paper introduces a new subspace based cross-domain recognition algorithm. We construct compact joint subspace independently for each class, which covers both source and target domains. It carries the information not only about the intrinsic characteristics of the considered class, but also about the specificity for each domain. Classifiers are trained on these compact joint subspaces. The proposed algorithm has been evaluated on two widely used datasets. Comparison results show that the proposed algorithm outperforms several existing methods on both datasets.

5. REFERENCES

[1] A. Torralba and A.A Efros, "Unbiased look at dataset bias," in *CVPR*, 2011.

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

[3] H. Daume III, "Frustratingly easy domain adaptation," in *ACL*, 2007.

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

[5] L. Duan, I.W Tsang, and D. Xu, "Domain transfer multiple kernel learning," *IEEE TPAMI*, vol. 34, no. 3, pp. 465–479, 2012.

[6] S. Shekhar, V. M. Patel, H. V. Nguyen, and R. Chellappa, "Generalized domain-adaptive dictionaries," in *CVPR*, 2013.

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

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

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

[10] J. Ni, Q. Qiu, and R. Chellappa, "Subspace interpolation via dictionary learning for unsupervised domain adaptation," in *CVPR*, 2013.

[11] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *ICCV*, 2013.

[12] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE TPAMI*, vol. 35, no. 11, pp. 2765–2781, 2013.

[13] R. Basri and D. Jacobs, "Lambertian reflection and linear subspaces," *IEEE TPAMI*, vol. 25, no. 3, pp. 218–233, 2003.

[14] J.J. DiCarlo, D. Zoccolan, and N.C. Rust, "How does the brain solve visual object recognition?," *Neuron*, vol. 73, no. 3, pp. 415–434, 2012.

[15] V. Zografos, L. Ellisy, and R. Mester, "Discriminative subspace clustering," in *CVPR*, 2013.

[16] J. Hamm and D. D. Lee, "Grassmann discriminant analysis: a unifying view on subspace-based learning," in *ICML*, 2008.

[17] D. Zhai, H. Chang, Y. Zhen, X. Liu, X. Chen, and W. Gao, "Parametric local multimodal hashing for cross-view similarity search," in *IJCAI*, 2013.

[18] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[19] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *ICML*, 2003.

[20] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speededup robust features (surf)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 1373–1396, 2008.

[21] Y. Shi and F. Sha, "Information-theoretical learning of discriminative clusters for unsupervised domain adaptation," in *ICML*, 2012.

[22] M. Yang, L. Zhang, X. Feng, and D. Zhang, "Fisher discrimination dictionary learning for sparse representation," in *ICCV*, 2011.

[23] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *ACL*, 2007.

[24] S.J. Pan, I.W Tsang, J.T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE TNN*, vol. 99, pp. 1–12, 2009.

[25] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *EMNLP*, 2006.

[26] J. Huang, A.J. Smola, A. Gretton, K.M. Borgwardt, and B. Scholkopf, "Correcting sample selection bias by unlabeled data," in *NIPS*, 2007.

[27] B. Gong, K. Grauman, and F. Sha, "Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation," in *ICML*, 2013.