Domain Adaptation via Transfer Component Analysis

Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang, *Fellow, IEEE*

***Abstract*— Domain adaptation allows knowledge from a source domain to be transferred to a different but related target domain. Intuitively, discovering a *good* feature representation across domains is crucial. In this paper, we first propose to find such a representation through a new learning method, *transfer component analysis* (TCA), for domain adaptation. TCA tries to learn some *transfer components* across domains in a reproducing kernel Hilbert space using maximum mean miscrepancy. In the subspace spanned by these *transfer components*, data properties are preserved and data distributions in different domains are close to each other. As a result, with the new representations in this subspace, we can apply standard machine learning methods to train classifiers or regression models in the source domain for use in the target domain. Furthermore, in order to uncover the knowledge hidden in the relations between the data labels from the source and target domains, we extend TCA in a semisupervised learning setting, which encodes label information into *transfer components* learning. We call this extension *semi-supervised TCA*. The main contribution of our work is that we propose a novel dimensionality reduction framework for reducing the distance between domains in a latent space for domain adaptation. We propose both unsupervised and semisupervised feature extraction approaches, which can dramatically reduce the *distance* between domain distributions by projecting data onto the learned *transfer components*. Finally, our approach can handle large datasets and naturally lead to out-of-sample generalization. The effectiveness and efficiency of our approach are verified by experiments on five toy datasets and two real-world applications: cross-domain indoor WiFi localization and cross-domain text classification.**

***Index Terms*— Dimensionality reduction, domain adaptation, Hilbert space embedding of distributions, transfer learning.**

I. Introduction

DOMAIN adaptation aims at solving a learning problem in the target domain by utilizing training data in the source domain, even when these domains may have different distributions. This is an important learning problem because labeled data are often difficult to come by, making it desirable to make the best use of any related data available. For example, in indoor WiFi localization which requires regression learning, the labeled training data are difficult and expensive to obtain [1]. Moreover, once calibrated, these data can be easily outdated because the WiFi signal strength may be a function of many dynamic factors including time, device, and space [2]. To reduce the recalibration effort, we might want to adapt a localization model trained in one time period (the source domain) for a new time period (the target domain), or to adapt the localization model trained on one mobile device (the source domain) for a new mobile device (the target domain).

Domain adaptation can be considered as a special setting of transfer learning which aims at transferring shared knowledge across different but related tasks or domains [3]–[5]. A major computational problem in domain adaptation is how to reduce the difference between the distributions of the source and target domain data. Intuitively, discovering a *good* feature representation across domains is crucial [3], [6]. A good feature representation should be able to reduce the difference in distributions between domains as much as possible, while at the same time preserving important properties (such as geometric properties, statistical properties, or side information [7]) of the original data, especially for the target domain data. Recently, several approaches have been proposed to learn a common feature representation for domain adaptation [8]–[10]. Daumé III [8] designed a heuristic kernel to augment features for solving some specific domain adaptation problems in natural language processing. Blitzer *et al*. [9] proposed the structural correspondence learning (SCL) algorithm, motivated from [11], to induce correspondences among features from the different domains. This method depends on the heuristic selections of pivot features appearing frequently in both domains. Although it is experimentally shown that SCL can reduce the difference between domains based on the $\mathcal{A}$-distance measure [6], the heuristic criterion of pivot feature selection may be sensitive to different applications.

Most previous feature-based domain adaptation methods do not minimize the distance in distributions between domains explicitly. Recently, von Bünau *et al*. [12] proposed stationary subspace analysis (SSA) to match distributions in a latent space. However, SSA is focused on the identification of a stationary subspace, without considering the preservation of properties such as data variance in the subspace. Pan *et al*. [10] proposed a new dimensionality reduction method called maximum mean discrepancy embedding (MMDE) for domain

Manuscript received January 15, 2010; revised October 14, 2010; accepted October 31, 2010. Date of publication November 18, 2010; date of current version February 9, 2011. The work of S. J. Pan and Q. Yang was supported in part by Hong Kong Research Grants Council/National Natural Science fund under Grant NHKUST 624/09. The work of I. W. Tsang was supported in part by Singapore Nanyang Technological University Academic Research Fund Tier-1 Research under Grant RG15/08 and State Emergency Response Commission under Grant 102 1580034. The work of J. T. Kwok was supported in part by Hong Kong Competitive Earmarked Research Grants under Project 615209.

S. J. Pan is with the Institute of Infocomm Research, 138632, Singapore (e-mail: sinnocat@gmail.com).

I. W. Tsang is with the School of Computer Engineering, Nanyang Technological University, 639798, Singapore (e-mail: ivortsang@ntu.edu.sg).

J. T. Kwok and Q. Yang are with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong (e-mail: jamesk@cse.ust.hk; qyang@cse.ust.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2010.2091281

adaptation. MMDE aims at learning a shared latent space underlying the domains where distance between distributions can be reduced while the data variance can be preserved. However, MMDE suffers from two major limitations: 1) MMDE is transductive, and does not generalize to out-of-sample patterns [13], and 2) MMDE learns the latent space by solving a *semi-definite program* (SDP), which is computationally expensive.

In this paper, we propose a new feature extraction approach, called *transfer component analysis* (TCA), for domain adaptation. It tries to learn a set of common *transfer components* underlying both domains such that the difference in data distributions of the different domains, when projected onto this subspace, can be dramatically reduced and data properties can be preserved. Standard machine learning methods can then be used in this subspace to train classification or regression models across domains. More specifically, if two domains are related to each other, there may exist several common components (or latent variables) underlying them. Some of these may cause the data distributions between domains to be different, while others may not. Meanwhile, some of these components may capture the intrinsic structure or discriminative information underlying the original data, while others may not. Hence, our goal is to discover those components that do not cause distribution change very much across the domains and can well preserve the structure or task-relevant information of the original data.

Our main contribution is on proposing a novel dimensionality reduction method to reduce the distance between domains via projecting data onto a learned transfer subspace. Once the subspace is found, one can use *any* method for subsequent classification, regression, and clustering. Furthermore, TCA and its semisupervised extension SSTCA are much more efficient than MMDE and can handle the out-of-sample extension problem [13]. The rest of this paper is organized as follows. In Section II, we first introduce the domain adaptation problem and traditional dimensionality reduction methods and describe the Hilbert space embedding for distances and dependence measure between distributions. Our proposed feature extraction methods for domain adaptation are presented in Sections III and IV. In Section V, we conduct a series of experiments on some toy datasets and two real-world application problems to verify the effectiveness and efficiency of the proposed methods. Finally, we conclude our work in Section VI.

II. Previous Works and Preliminaries

A. Domain Adaptation

We consider a domain as consisting of two main components: a feature space of inputs $\mathcal{X}$ and a marginal probability distribution of inputs $P(X)$, where $X = \{x_1, \ldots, x_n\} \in \mathcal{X}$ is a set of learning samples. For example, if our learning task is document classification, and each term is taken as a binary feature, then $\mathcal{X}$ is the space of all document vectors. In general, if two domains are different, they may have different feature spaces or different marginal probability distributions. In this paper, we focus on the setting where there are only one source and one target domain sharing the same feature space. We also assume that some labeled data $\mathcal{D}_S$ are available in the source domain, while only unlabeled data $\mathcal{D}_T$ are available in the target domain. More specifically, let the source domain data be $\mathcal{D}_S = \{(x_{S_1}, y_{S_1}), \ldots, (x_{S_{n_1}}, y_{S_{n_1}})\}$, where $x_{S_i} \in \mathcal{X}$ is the input and $y_{S_i} \in \mathcal{Y}$ is the corresponding output. Similarly, let the target domain data be $\mathcal{D}_T = \{x_{T_1}, \ldots, x_{T_{n_2}}\}$, where the input x_{T_i} is also in $\mathcal{X}$.

Let $\mathcal{P}(X_S)$ and $\mathcal{Q}(X_T)$ (or $\mathcal{P}$ and $\mathcal{Q}$ in short) be the marginal distributions of $X_S = \{x_{S_i}\}$ and $X_T = \{x_{T_i}\}$ from the source and target domains, respectively. In general, $\mathcal{P}$ and $\mathcal{Q}$ can be different. Our task is to predict the labels y_{T_i}s corresponding to inputs x_{T_i}s in the target domain. The key assumption in most domain adaptation methods is that $\mathcal{P} \neq \mathcal{Q}$, but $P(Y_S|X_S) = P(Y_T|X_T)$.

The problem of covariate shift adaptation is also related to domain adaption. To address this problem, *importance* reweighting is a major technique [14]–[18]. Huang *et al*. [14] proposed a kernel-based method, known as *kernel mean matching* (KMM) to reweight instances in a reproducing kernel Hilbert space (RKHS). Sugiyama *et al.* [15] proposed another importance reweighting algorithm, known as *Kullback–Leibler importance estimation procedure* (KLIEP), which is integrated with cross validation to perform model selection automatically. Bickel *et al*. [16] proposed to integrate the distribution correcting process into a kernelized logistic regression. Kanamori *et al*. [17] proposed a method called *unconstrained least-squares importance fitting* (uLSIF) to estimate the importance efficiently by formulating the direct importance estimation problem as a least-squares function fitting problem. The main difference between these methods and our proposed method is that we aim to match data distributions between domains in a latent space, where data properties can be preserved, instead of matching them in the original feature space. Recently, Sugiyama *et al*. [18] further extended the uLSIF algorithm by estimating importance in a nonstationary subspace, which performs well even when the dimensionality of the data domains is high. However, this method is focused on estimating the *importance* in a latent space instead of learning a latent space for adaptation. Note that, besides covariate shift adaptation, importance estimation techniques have also been applied to various applications, such as independent component analysis [19], outlier detection [20], and change-point detection [21].

Besides reweighting methods, von Bünau *et al*. [12] proposed to match distributions in a latent space. More specifically, they theoretically studied the conditions under which a stationary space can be identified from a multivariate time series. They also proposed the SSA procedure to find stationary components by matching the first two moments of the data distributions in different epochs. However, SSA is focused on how to identify a stationary subspace without considering how to preserve data properties in the latent space. As a result, SSA may map the data to some noisy factors which are stationary across domains but completely irrelevant to the target supervised task. Then classifiers trained on the new representations learned by SSA may not get good performance for domain adaptation. We will show a motivating example in Section III.

B. Hilbert Space Embedding of Distributions

1) Maximum Mean Discrepancy (MMD): Given samples $X = \{x_i\}$ and $Y = \{y_i\}$ drawn from two distributions, there exist many criteria [such as the Kullback–Leibler (KL) divergence] that can be used to estimate their distance. However, many of these estimators are parametric or require an intermediate density estimate. Recently, a nonparametric distance estimate was designed by embedding distributions in an RKHS [22]. Gretton *et al*. [23] introduced the MMD for comparing distributions based on the corresponding RKHS distance. Let the kernel-induced feature map be ϕ. The empirical estimate of MMD between $\{x_1, \ldots, x_{n_1}\}$ and $\{y_1, \ldots, y_{n_2}\}$ is $\mathrm{MMD}(X, Y) = \|1/n_1 \sum_{i=1}^{n_1} \phi(x_i) - 1/n_2 \sum_{i=1}^{n_2} \phi(y_i)\|_{\mathcal{H}}^2$ where $\|\cdot\|_{\mathcal{H}}$ is the RKHS norm. Therefore, the distance between two distributions is simply the distance between the two mean elements in a RKHS. It can be shown [22] that, when the RKHS is universal, MMD will asymptotically approach zero if and only if the two distributions are the same.

2) Hilbert–Schmidt Independence Criterion (HSIC): Related to the MMD, the HSIC [24] is a simple yet powerful nonparametric criterion for measuring the dependence between the sets X and Y. As its name implies, it computes the Hilbert–Schmidt norm of a cross-covariance operator in the RKHS. An (biased) empirical estimate can be easily obtained from the corresponding kernel matrices, as $\mathrm{HSIC}(X, Y) = (1/(n-1)^2)\mathrm{tr}(HKHK_{yy})$ where K, K_{yy} are kernel matrices defined on X and Y, respectively, $H = I - (1/n)\mathbf{1}\mathbf{1}^\top$ is the centering matrix, and n is the number of samples in X and Y. Similar to MMD, it can also be shown that if the RKHS is universal, HSIC asymptotically approaches zero if and only if X and Y are independent [25]. Conversely, a large HSIC value suggests strong dependence.

C. Embedding Using HSIC

In embedding or dimensionality reduction, it is often desirable to preserve the local data geometry while at the same time maximally align the embedding with available side information (such as labels). For example, in colored maximum variance unfolding (colored MVU) [7], the local geometry is captured in the form of local distance constraints on the target embedding K, while the alignment with the side information (represented as kernel matrix K_{yy}) is measured by the HSIC criterion. Mathematically, this leads to the following SDP:

$$\max_{K \succeq 0} \mathrm{tr}(HKHK_{yy}) \text{ subject to constraints on } K. \quad (1)$$

In particular, (1) reduces to MVU [26] when no side information is given (i.e., $K_{yy} = I$).

III. TCA

As mentioned in Section II-A, most domain adaptation methods assume that $\mathcal{P} \neq \mathcal{Q}$, but $P(Y_S|X_S) = P(Y_T|X_T)$. However, in many real-world applications, the conditional probability $P(Y|X)$ may also change across domains due to noisy or dynamic factors underlying the observed data. In this paper, we use the weaker assumption that $\mathcal{P} \neq \mathcal{Q}$, but there exists a transformation ϕ such that $P(\phi(X_S)) \approx P(\phi(X_T))$ and $P(Y_S|\phi(X_S)) \approx P(Y_T|\phi(X_T))$. Standard supervised learning methods can then be applied on the mapped source domain data $\phi(X_S)$, together with the corresponding labels Y_S, to train models for use on the mapped target domain data $\phi(X_T)$.

A key issue is how to find this transformation ϕ. Since we have no labeled data in the target domain, ϕ cannot be learned by directly minimizing the distance between $P(Y_S|\phi(X_S))$ and $P(Y_T|\phi(X_T))$. Here, we propose to learn ϕ such that: 1) the distance between the marginal distributions $P(\phi(X_S))$ and $P(\phi(X_T))$ is small, and 2) $\phi(X_S)$ and $\phi(X_T)$ preserve important properties of X_S and X_T. We then assume that such a ϕ satisfies $P(Y_S|\phi(X_S)) \approx P(Y_T|\phi(X_T))$. We believe that domain adaptation under this assumption is more realistic, though also more challenging. Finally, a classifier f trained on $\phi(X_S)$ and Y_S is used to make predictions on $\phi(X_T)$.

A. Minimizing Distance Between $P(\phi(X_S))$ and $P(\phi(X_T))$

Assume that ϕ is the feature map induced by a universal kernel. As shown in Section II-B.1, the distance between two distributions $\mathcal{P}$ and $\mathcal{Q}$ can be empirically measured by the distance between the empirical means of the two domains $\mathrm{Dist}(X'_S, X'_T) = \left\|1/n_1 \sum_{i=1}^{n_1} \phi(x_{S_i}) - 1/n_2 \sum_{i=1}^{n_2} \phi(x_{T_i})\right\|_{\mathcal{H}}^2$. Therefore, a desired nonlinear mapping ϕ can be found by minimizing this quantity. However, ϕ is usually highly nonlinear and a direct optimization of minimizing the quantity with respect to ϕ can get stuck in poor local minima.

1) MMDE: Instead of finding the nonlinear transformation ϕ explicitly, we first revisit a dimensionality reduction-based domain adaptation method called MMDE [10]. It embeds both the source and target domain data into a shared low-dimensional latent space using a nonlinear mapping ϕ, and then learns the corresponding kernel matrix K by solving an SDP. Specifically, let the Gram matrices defined on the source domain, target domain, and cross-domain data in the embedded space be $K_{S,S}$, $K_{T,T}$, and $K_{S,T}$, respectively. The key idea is to learn

$$K = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S} & K_{T,T} \end{bmatrix} \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)} \quad (2)$$

i.e., the kernel matrix defined on all the data, by minimizing the distance (measured w.r.t. the MMD) between the projected source and target domain data while maximizing the embedded data variance. By virtue of the kernel trick, it can be shown that the MMD distance in Section III-A can be written as $\mathrm{tr}(KL)$, where $K = [\phi(x_i)^\top \phi(x_j)]$, and $L_{ij} = 1/n_1^2$ if $x_i, x_j \in X_S$, else $L_{ij} = 1/n_2^2$ if $x_i, x_j \in X_T$, otherwise, $L_{ij} = -(1/n_1 n_2)$.

The objective function of MMDE can then be written as

$$\max_{K \succeq 0} \mathrm{tr}(KL) - \lambda \mathrm{tr}(K) \text{ subject to constraints on } K \quad (3)$$

where the first term in the objective minimizes the distance between distributions, while the second term maximizes the variance in the feature space, and $\lambda \geq 0$ is a tradeoff parameter.

However, there are several limitations of MMDE. First, it is transductive and cannot generalize to unseen patterns. Second, the resultant kernel learning problem has to be solved

by expensive SDP solvers. Finally, in order to construct low-dimensional representations of X'_S and X'_T, the obtained K has to be further post-processed by PCA [10]. This may discard potentially useful information in K.

2) Parametric Kernel Map for Unseen Patterns: In this section, we propose an efficient framework to find the nonlinear mapping ϕ based on kernel feature extraction. It avoids the use of SDP and thus its high computational burden. Moreover, the learned kernel can be generalized to out-of-sample patterns. Besides, instead of using a two-step approach as in MMDE, we propose a unified kernel learning method which utilizes an explicit low-rank representation.

First, note that the kernel matrix K in (2) can be decomposed as $K = (KK^{-1/2})(K^{-1/2}K)$, which is often known as the empirical kernel map [27]. Consider the use of a matrix $\widetilde{W} \in \mathbb{R}^{(n_1+n_2)\times m}$ that transforms the empirical kernel map features to an m-dimensional space (where $m \ll n_1 + n_2$). The resultant kernel matrix is then

$$\widetilde{K} = (KK^{-1/2}\widetilde{W})(\widetilde{W}^\top K^{-1/2}K) = KWW^\top K \tag{4}$$

where $W = K^{-1/2}\widetilde{W}$. In particular, the corresponding kernel evaluation between any two patterns x_i and x_j is $\widetilde{k}(x_i, x_j) = k_{x_i}^\top WW^\top k_{x_j}$ where $k_x = [k(x_1, x), \ldots, k(x_{n_1+n_2}, x)]^\top \in \mathbb{R}^{n_1+n_2}$. Hence, this kernel $\widetilde{k}$ facilitates a readily parametric form for out-of-sample kernel evaluations.

On using the definition of $\widetilde{K}$ in (4), the MMD distance between the empirical means of the two domains X'_S and X'_T can be rewritten as

$$\text{Dist}(X'_S, X'_T) = \text{tr}((KWW^\top K)L) = \text{tr}(W^\top KLKW). \tag{5}$$

In minimizing (5), a regularization term $\text{tr}(W^\top W)$ is usually needed to control the complexity of W. As will be shown later, this regularization term can also avoid the rank deficiency of the denominator in the generalized eigenvalue decomposition.

B. Preserving Properties of X_S and X_T

In domain adaptation, learning the transformation ϕ by only minimizing the distance between $P(\phi(X_S))$ and $P(\phi(X_T))$ may not be enough. Fig. 1(a) shows a simple 2-D example, where the source domain data is in red and the target domain data is in blue. For both domains, x_1 is the discriminative direction that separates the positive and negative samples, while x_2 is a noisy dimension with small variance. By focusing only on minimizing the distance between $P(\phi(X_S))$ and $P(\phi(X_T))$, one would select the noisy component x_2, which however is completely irrelevant to the target supervised task.

Hence, besides reducing the distance between the two marginal distributions, ϕ should also preserve data properties that are useful for the target supervised learning task. An obvious choice is to maximally preserve the data variance, as performed by PCA and KPCA. Note from (4) that the embedding of the data in the latent space is $W^\top K$, where the ith column $[W^\top K]_i$ provides the embedding coordinates of x_i. Hence, the variance of the projected samples is $W^\top KHKW$, where $H = I_{n_1+n_2} - (1/n_1 + n_2)\mathbf{1}\mathbf{1}^\top$ is the centering matrix, $\mathbf{1} \in \mathbb{R}^{n_1+n_2}$ is the column vector with all 1's, and $I_{n_1+n_2} \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$ is the identity matrix.

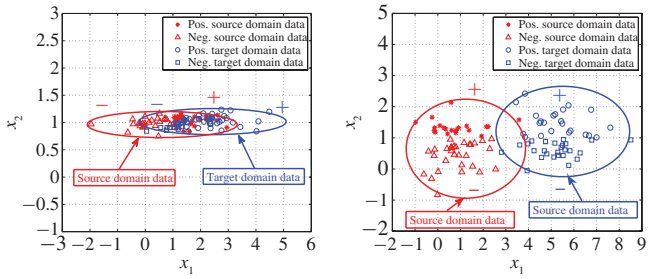

Fig. 1. Motivating examples for the TCA formulation. (a) Only minimizing the distance between $P(\phi(X_S))$ and $P(\phi(X_T))$. (b) Only maximizing the data variance.

However, focusing only on the data variance is again not desirable in domain adaptation. An example is shown in Fig. 1(b), where the direction with the largest variance (x_1) cannot be used to reduce the distance of distributions across domains and is not useful in boosting the performance for domain adaptation.

C. Unsupervised TCA

Combining the observations in Sections III-A and III-B, we develop a new dimensionality reduction method for domain adaptation such that in the latent space spanned by the learned components, the variance of the data can be preserved as much as possible and the distance between different distributions across domains can be reduced. The kernel learning problem then becomes

$$\begin{aligned} \min_W \quad & \text{tr}(W^\top KLKW) + \mu\ \text{tr}(W^\top W) \\ \text{s.t.} \quad & W^\top KHKW = I_m \end{aligned} \tag{6}$$

where $\mu > 0$ is a tradeoff parameter, and $I_m \in \mathbb{R}^{m\times m}$ is the identity matrix. For notational simplicity, we will drop the subscript m from I_m in the sequel.

Though this optimization problem involves a nonconvex norm constraint $W^\top KHKW = I$, it can still be solved efficiently by the following trace optimization problem:

Proposition 1: Problem (6) can be reformulated as

$$\max_W \ \text{tr}((W^\top(KLK + \mu I)W)^{-1}W^\top KHKW). \tag{7}$$

Proof: The Lagrangian of (6) is

$$\text{tr}(W^\top(KLK + \mu I)W) - \text{tr}((W^\top KHKW - I)Z) \tag{8}$$

where Z is a diagonal matrix containing Lagrange multipliers. Setting the derivative of (8) w.r.t. W to zero, we have $(KLK + \mu I)W = KHKWZ$. Multiplying both sides on the left by W^T, and then on substituting it into (8), we obtain $\min_W \ \text{tr}((W^\top KHKW)^\dagger W^\top(KLK + \mu I)W)$. Since the matrix $KLK + \mu I$ is nonsingular, we obtain an equivalent trace maximization problem (7). ■

Similar to kernel Fisher discriminant analysis [28], the W solutions in (7) are the m leading eigenvectors of $(KLK + \mu I)^{-1}KHK$, where $m \le n_1 + n_2 - 1$. In the sequel, this will be referred to as TCA.

IV. SSTCA

As discussed in [6], a good representation should: 1) reduce the distance between the distributions of the source and target domain data, and 2) minimize the empirical error on the labeled data in the source domain. However, the unsupervised TCA proposed in Section III-C does not consider the label information in learning the components. Moreover, in many real-world applications (such as WiFi localization), there exists an intrinsic low-dimensional manifold underlying the high-dimensional observations. The effective use of manifold information is an important component in many semisupervised learning algorithms [29].

In this section, we extend the *unsupervised* TCA in Section III-C to the semisupervised learning setting. Motivated by the kernel target alignment [30], [31], a representation that maximizes its dependence with the data labels may lead to better generalization performance. Hence, we can maximize the label dependence instead of minimizing the empirical error (Section IV-A.2). Moreover, we encode the manifold structure into the embedding learning so as to propagate label information from the labeled (source domain) data to the unlabeled (target domain) data (Section IV-A.3). Note that in traditional semisupervised learning settings, the labeled and unlabeled data are from the same domain. However, in the context of domain adaptation here, the labeled and unlabeled data are from different domains.

A. Optimization Objectives

In this section, we delineate three desirable properties for this semisupervised embedding, namely: 1) maximal alignment of distributions between the source and target domain data in the embedded space; 2) high dependence on the label information; and 3) preservation of the local geometry.

1) Objective 1: Distribution Matching: As in the *unsupervised* TCA, our first objective is to minimize the MMD (5) between the source and target domain data in the embedded space.

2) Objective 2: Label Dependence: Our second objective is to maximize the dependence (measured w.r.t. HSIC) between the embedding and labels. Recall that, while the source domain data are fully labeled, the target domain data are unlabeled. We propose to maximally align the embedding [which is represented by $\widetilde{K}$ in (4)] with

$$\tilde{K}_{yy} = \gamma K_l + (1-\gamma)K_v \tag{9}$$

where $\gamma \geq 0$. Here, $[K_l]_{ij} = k_{yy}(y_i, y_j)$ if $i, j \leq n_1$, otherwise $[K_l]_{ij} = 0$, serves to maximize label dependence on the labeled data, while $K_v = I$, serves to maximize the variance on both the source and target domain data, which is in line with MVU [26]. By substituting $\widetilde{K}$ (4) and $\tilde{K}_{yy}$ (9) into HSIC (Section II-B.2), our objective is thus to maximize

$$\mathrm{tr}(H(KWW^\top K)H\tilde{K}_{yy}) = \mathrm{tr}(W^\top KH\tilde{K}_{yy}HKW). \tag{10}$$

Note that γ is a tradeoff parameter that balances the label dependence and data variance terms. Intuitively, if there are sufficient labeled data in the source domain, the dependence between features and labels can be estimated more precisely via HSIC, and a large γ may be used. Otherwise, when there are only a few labeled data in the source domain and a large number of unlabeled data in the target domain, we may use a small γ. Empirically, simply setting $\gamma = 0.5$ works well on all the datasets. The sensitivity of the performance to γ will be studied in more detail in Sections V-B and V-C.

3) Objective 3: Locality Preserving: As reviewed in Sections II-C and III-A.1, colored MVU and MMDE preserve the local geometry of the manifold by enforcing distance constraints on the desired kernel matrix K. More specifically, let $\mathcal{N} = \{(x_i, x_j)\}$ be the set of sample pairs that are k-nearest neighbors of each other, and $d_{ij} = \|x_i - x_j\|$ be the distance between x_i, x_j in the original input space. For each (x_i, x_j) in $\mathcal{N}$, a constraint $K_{ii} + K_{jj} - 2K_{ij} = d_{ij}^2$ will be added to the optimization problem. Hence, the resultant SDP will typically have a very large number of constraints.

To avoid this problem, we make use of the locality preserving property of the manifold regularizer [32]. First, we construct a graph with the affinity $m_{ij} = \exp(-d_{ij}^2/2\sigma^2)$ if x_i is one of the k nearest neighbors of x_j, or vice versa. Let $M = [m_{ij}]$. The graph Laplacian matrix is $\mathcal{L} = D - M$, where D is the diagonal matrix with entries $d_{ii} = \sum_{j=1}^{n} m_{ij}$. Intuitively, if x_i, x_j are neighbors in the input space, the distance between the embedding coordinates of x_i and x_j should be small. Note that the data's embedding in $\mathbb{R}^m$ is $W^\top K$, where the ith column $[W^\top K]_i$ provides the embedding coordinates of x_i. Hence, our third objective is to minimize

$$\sum_{(i,j)\in\mathcal{N}} m_{ij} \left\| [W^\top K]_i - [W^\top K]_j \right\|^2 = \mathrm{tr}(W^\top K\mathcal{L}KW). \tag{11}$$

B. Formulation and Optimization Procedure

Combining all three objectives, we thus want to find a W that maximizes (10) while simultaneously minimizing (5) and (11). The final optimization problem can be written as

$$\begin{aligned} &\min_W \mathrm{tr}(W^\top KLKW) + \mu\ \mathrm{tr}(W^\top W) + \frac{\lambda}{n^2}\ \mathrm{tr}(W^\top K\mathcal{L}KW) \\ &\text{s.t.}\ \ W^\top KH\widetilde{K}_{yy}HKW = I \end{aligned} \tag{12}$$

where $\lambda \geq 0$ is another tradeoff parameter, and $n^2 = (n_1 + n_2)^2$ is a normalization term. For simplicity, we use λ to denote λ/n^2 in the rest of this paper. Similar to the *unsupervised* TCA, (12) can be formulated as the following trace problem:

$$\max_W \mathrm{tr}\left\{\left(W^\top K(L + \lambda\mathcal{L})KW + \mu I\right)^{-1}\left(W^\top KH\widetilde{K}_{yy}HKW\right)\right\}.$$

We call this semisupervised transfer component analysis (SSTCA). It is Well known that it can be solved by eigen-decomposing $(K(L + \lambda\mathcal{L})K + \mu I)^{-1}KH\widetilde{K}_{yy}HK$.

The procedure for both the unsupervised and semisupervised TCA is summarized in Algorithm 1.

C. Computational Issues

The kernel learning algorithm for domain adaptation in [10] relies on SDPs. As there are $O((n_1+n_2)^2)$ variables in $\widetilde{K}$, the overall time complexity is $O((n_1+n_2)^{6.5})$ [33]. This becomes

Algorithm 1: TCA

Input: Source domain data set $\mathcal{D}_S = \{(x_{S_i}, y_{src_i})\}_{i=1}^{n_1}$, and target domain data set $\mathcal{D}_T = \{x_{T_j}\}_{j=1}^{n_2}$.

Output: Transformation matrix W.

1: Construct kernel matrix K from $\{x_{S_i}\}_{i=1}^{n_1}$ and $\{x_{T_j}\}_{j=1}^{n_2}$ based on (2), matrix L from (3), and centering matrix H.

2: (Unsupervised TCA) Eigendecompose the matrix $(KLK + \mu I)^{-1}KHK$ and select the m leading eigenvectors to construct the transformation matrix W.

3: (Semisupervised TCA) Eigendecompose matrix $(K(L + \lambda)\mathcal{L}K + \mu I)^{-1}KH\widetilde{K}_{yy}HK$ and select the m leading eigenvectors to construct the transformation matrix W.

4: **return** transformation matrix W.

computationally prohibitive even for small-sized problems. In contrast, our proposed kernel learning method requires only a simple and efficient eigenvalue decomposition. This takes only $O(m(n_1 + n_2)^2)$ time when m nonzero eigenvectors are to be extracted [34].

V. Experiments

In this section, we first verify the motivations of our proposed methods for domain adaptation on some toy datasets.

A. Synthetic Data

As discussed in Section II-A, the optimization objective needs to include a number of criteria. In this section, we perform experiments to demonstrate the effectiveness of TCA/SSTCA in learning a 1-D latent space from the 2-D data. For TCA, we use the linear kernel on inputs, and fix $\mu = 1$. For SSTCA, we use the linear kernel on both inputs and outputs, and fix $\mu = 1$, $\gamma = 0.5$.

1) Only Minimizing Distance between Distributions: As discussed in Section III-B, it is not desirable to learn the transformation ϕ by only minimizing the distance between the marginal distributions $P(\phi(X_S))$ and $P(\phi(X_T))$. Here, we illustrate this by using the synthetic data from the example in Fig. 1(a) [which is also reproduced in Fig. 2(a)]. We compare TCA with the method of SSA [12], which is an empirical method to find an identical stationary latent space of the source and target domain data.

As can be seen from Fig. 2(b), the distance between distributions of different domain data in the 1-D space learned by SSA is small. However, the positive and negative samples are overlapped together in this latent space, which is not useful for making predictions on the mapped target domain data. On the other hand, as can be seen from Fig. 2(c), though the distance between distributions of different domain data in the latent space learned by TCA is larger than that learned by SSA, the two classes are now more separated. We further apply the one-nearest-neighbor (1-NN) classifier to make predictions on the target domain data in the original 2-D space, and latent spaces learned by SSA and TCA. As can be seen from Fig. 2(a)–(c), TCA leads to significantly better accuracy than SSA.

2) Only Maximizing the Data Variance: As discussed in Section III-B, learning the transformation ϕ by only maximizing the data variance may not be useful in domain adaptation. Here, we reproduce Fig. 1(b) in Fig. 2(d). As can be seen from Fig. 2(e) and (f), the variance of the mapped data in the 1-D space learned by PCA is very large. However, the distance between the mapped data across different domains is still large and the positive and negative samples are overlapped together in the latent space, which is not useful for domain adaptation. On the other hand, though the variance of the mapped data in the 1-D space learned by TCA is smaller than that learned by PCA, the distance between different domain data in the latent space is reduced and the positive and negative samples are more separated in the latent space.

3) Label Information: In this experiment, we demonstrate the advantage of using label information in the source domain data to improve classification performance [Fig. 2(g)]. Since the focus is not on locality preserving, we set the λ in SSTCA to zero. Consequently, the difference between SSA and SSTCA is in the use of label information. As can be seen from Fig. 2(h), the positive and negative samples overlap significantly in the latent space learned by TCA. On the other hand, with the use of label information, the positive and negative samples are more separated in the latent space learned by SSTCA [Fig. 2(i)], and thus classification also becomes easier.

However, in some applications, it may be possible that the discriminative direction of the source domain data is quite different from that of the target domain data. An example is shown in Fig. 2(j). In this case, encoding label information from the source domain (as SSTCA does) may not help or even hurt the classification performance as compared to the unsupervised TCA. As can be seen from Fig. 2(k) and (l), positive and negative samples in the target domain are more separated in the latent space learned by TCA than in that learned by SSTCA.

In summary, when the discriminative directions across different domains are similar, SSTCA can outperform TCA by encoding label information into the embedding learning. However, when the discriminative directions across different domains are different, SSTCA may not improve the performance or even performs worse than TCA. Nevertheless, compared to nonadaptive methods, both SSTCA and TCA can obtain better performance.

4) Manifold Information: In this experiment, we demonstrate the advantage of using manifold information to improve classification performance. Both the source and domain data have the well-known two-moon manifold structure [29] [Fig. 2(m)]. SSTCA is used with and without Laplacian smoothing [by setting λ in (12) to 1000 and 0, respectively]. As can be seen from Fig. 2(n) and (o), Laplacian smoothing can indeed help improve classification performance when the manifold structure is available underlying the observed data.

B. Cross-Domain Indoor WiFi Localization

With the increasing availability of 802.11 WiFi networks in cities and buildings, locating and tracking a user or cargo

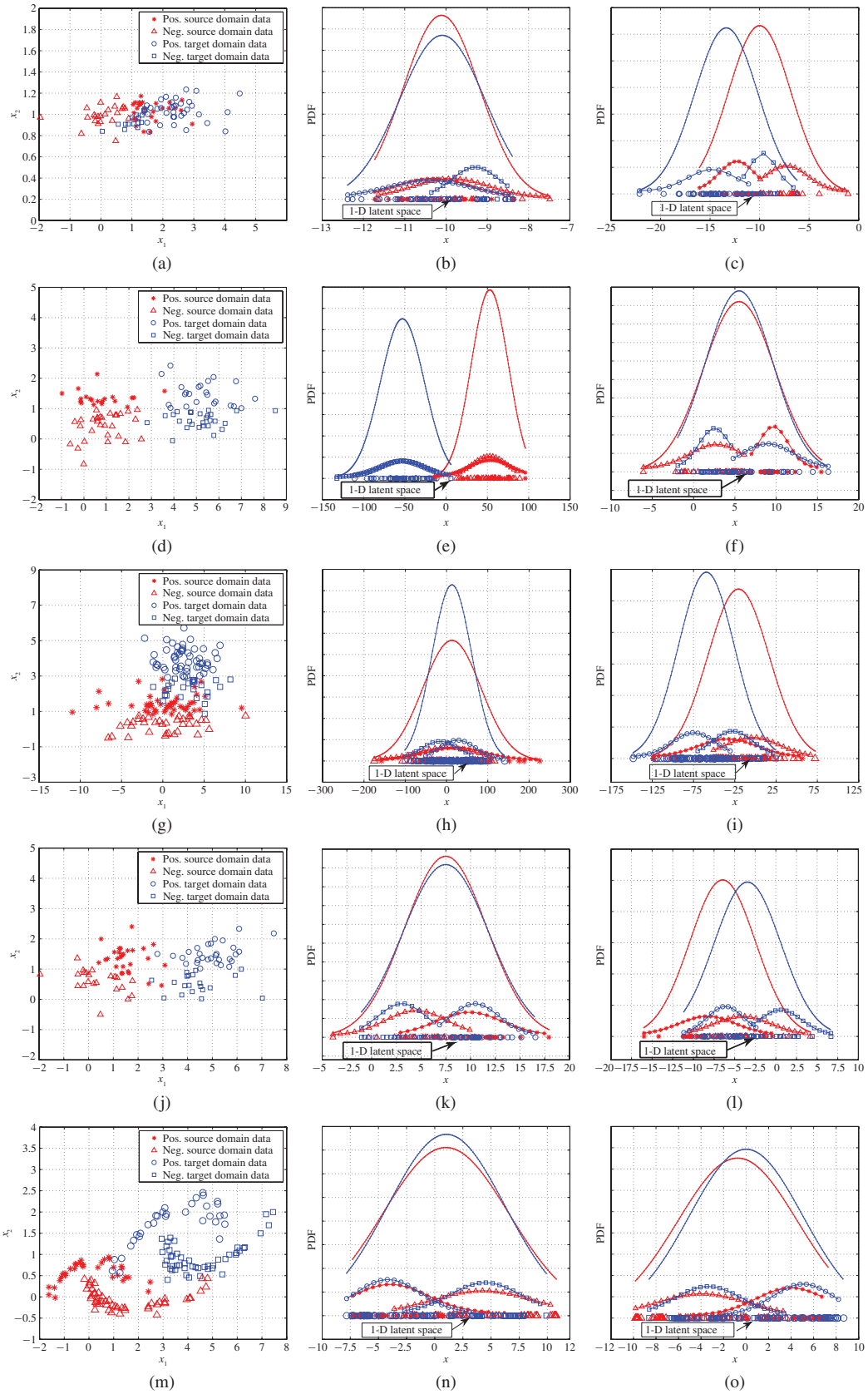

Fig. 2. Illustrations of the proposed TCA and SSTCA on five synthetic datasets. The leftmost column shows data in the original 2-D input space, while the other columns show the projected data in the 1-D latent spaces learned by different methods. Accuracy of the 1-NN classifier in the original input/latent space is shown inside brackets. (a) Dataset 1 (acc: 82%). (b) 1-D projection by SSA (acc: 60%). (c) 1-D projection by TCA (acc: 86%). (d) Dataset 2 (accuracy: 50%). (e) 1-D projection by PCA (acc: 48%). (f) 1-D projection by TCA (acc: 82%). (g) Data set 3 (acc: 69%). (h) 1-D projection by TCA (acc: 56%). (i) 1-D projection by SSTCA (acc: 79%). (j) Dataset 4 (accuracy: 60%). (k) 1-D projection by TCA (acc: 90%). (l) 1-D projection by SSTCA (acc: 68%). (m) Dataset 5 (acc: 70%). (n) 1-D projection by SSTCA without Laplacian smoothing (acc: 83%). (o) 1-D projection by SSTCA with Laplacian smoothing (acc: 91%).

with wireless signal strength or received signal strength (RSS) is becoming a reality [1], [2]. The objective of indoor WiFi localization is to estimate the location y_i of a mobile device based on the RSS values $x_i = (x_{i_1}, x_{i_2}, \ldots, x_{i_k})$ received from k access points, which periodically send out wireless signals to others. In the following, we consider the 2-D coordinates of a location, and indoor WiFi localization is intrinsically a regression problem. However, it is expensive to calibrate a localization model in a large environment. Moreover, the RSS values are noisy and can vary with time [2]. As a result, even in the same environment, the RSS data collected in one time period may differ from those collected in another. Hence, domain adaptation is necessary for indoor WiFi localization.

1) Experimental Setup: We use a public dataset from the 2007 IEEE ICDM Contest (the second task). This contains a few labeled WiFi data collected in time period T_1 (the source domain) and a large amount of unlabeled WiFi data collected in time period T_2 (the target domain). Here, "label" refers to the location information for which the WiFi data are received. WiFi data collected from different time periods are considered as different domains. The task is to predict the labels of the WiFi data collected in time period T_2. For more details on the dataset, readers may refer to the contest report article [2].

Denote the data collected in time period T_1 and time period T_2 by $\mathcal{D}_S$ and $\mathcal{D}_T$, respectively. In the experiments, we have $|\mathcal{D}_S| = 621$ and $|\mathcal{D}_T| = 3,128$. Furthermore, we randomly split $\mathcal{D}_T$ into $\mathcal{D}_T^u$ (the label information is removed in training) and $\mathcal{D}_T^o$. All the source domain data (621 instances in total) are used for training. As for the target domain data, 2328 patterns are sampled to form D_T^o, and a variable number of patterns are sampled from the rest 800 patterns to form D_T^u.

In the transductive evaluation setting, our goal is to learn a model from $\mathcal{D}_S$ and $\mathcal{D}_T^u$, and then evaluate the model on $\mathcal{D}_T^u$. In the out-of-sample evaluation setting, our goal is to learn a model from $\mathcal{D}_S$ and $\mathcal{D}_T^u$, and then evaluate the model on $\mathcal{D}_T^o$ (out-of-sample patterns). For each experiment, we repeat 10 times and then report the average performance using the average error distance (AED): $AED = \left(\sum_{(x_i,y_i)\in\mathcal{D}} |f(x_i) - y_i|\right)/N$. Here, x_i is a vector of RSS values, $f(x_i)$ is the predicted location, y_i is the corresponding ground truth location, while $\mathcal{D} = \mathcal{D}_T^u$ in the transductive setting, and $\mathcal{D} = \mathcal{D}_T^o$ in the out-of-sample evaluation setting.

The following methods will be compared. For parameter tuning of all methods, 50 labeled data are sampled from the source domain as a validation set. 1) Traditional regression models that do not perform domain adaptation. These include the (supervised) regularized least square regression (RLSR), which is trained on $\mathcal{D}_S$ only, and the (semisupervised) Laplacian RLSR (LapRLSR) [32], which is trained on both $\mathcal{D}_S$ and $\mathcal{D}_T^u$ but without considering the difference in distributions. 2) A traditional dimensionality reduction method: Kernel PCA (KPCA) [27]. It first learns a projection from both $\mathcal{D}_S$ and $\mathcal{D}_T^u$ via KPCA. RLSR is then applied on the projected $\mathcal{D}_S$ to learn a localization model. 3) Importance reweighting methods: KMM and KLIEP. They use both $\mathcal{D}_S$ and $\mathcal{D}_T^u$ to learn weights of the patterns in $\mathcal{D}_S$, and then train an RLSR model on the weighted data. Following [14], we set the ϵ parameter in KMM as $B/\sqrt{n_1}$, where n_1 is the number of training data in the source domain. For KLIEP, we use the *likelihood cross-validation* method in [15] to automatically select the kernel width. Preliminary results suggest that the final performance of KLIEP can be sensitive to the initialization of the kernel width. Thus, its initial value is also tuned on the validation set. 4) A state-of-the-art domain adaptation method. SCL [9][1] It learns a set of new cross-domain features from both $\mathcal{D}_S$ and $\mathcal{D}_T^u$, and then augments features on the source domain data in $\mathcal{D}_S$ with the new features. A RLSR model is then trained. 5) The proposed TCA and SSTCA. First, we apply TCA/SSTCA on both $\mathcal{D}_S$ and $\mathcal{D}_T^u$ to learn transfer components, and map data in $\mathcal{D}_S$ to the latent space. Finally, a RLSR model is trained on the projected source domain data. There are two parameters in TCA, kernel width[2] σ and parameter μ. We first set $\mu = 1$ and search for the best σ value (based on the validation set) in the range $[10^{-5}, 10^5]$. Afterwards, we fix σ and search for the best μ value in $[10^{-3}, 10^3]$. For SSTCA, we use linear kernel for k_{yy} in (9) on the labels, and there are four tunable parameters (σ, μ, λ, and γ). We set σ and μ in the same manner as TCA. Then, we set $\gamma = 0.5$ and search for the best λ value in $[10^{-6}, 10^6]$. Afterwards, we fix λ and search for γ in $[0, 1]$. 6) Methods that only perform distribution matching in a latent space: SSA [12] and TCAReduced, which replaces the constraint $W^\top KHKW = I$ in TCA by $W^\top W = I$. Hence, TCAReduced aims to find a transformation W that minimizes the distance between different distributions without maximizing the variance in the latent space. 7) A closely related dimensionality reduction method, MMDE. This is a state-of-the-art method on the ICDM-09 contest dataset [35].

2) Comparison to Dimensionality Reduction Methods: We first compare TCA and SSTCA with some dimensionality reduction methods, including KPCA, SSA and TCAReduced in the out-of-sample setting. The number of unlabeled patterns in $\mathcal{D}_T^u$ is fixed at 400, while the dimensionality of the latent space varies from 5 to 50.

Fig. 3(a) shows the results. As can be seen, TCA and SSTCA outperform all the other methods. Moreover, note that KPCA, though simple, can lead to significantly improved performance. This is because the WiFi data are highly noisy, and thus localization models learned in the denoised latent space can be more accurate than those learned in the original input space. However, as mentioned in Section V-A, KPCA can only denoise but cannot ensure that the distance between data distributions in the two domains is reduced. Thus, TCA performs better than KPCA. In addition, though TCAReduced and SSA aim to reduce distance between domains, they may lose important information of the original data in the latent space, which in turn may hurt performance of the target learning tasks. Thus, they do not obtain good performance. Finally, we observe that SSTCA obtains better performance than

[1]The pivot features are selected by mutual information while the number of pivots and other SCL parameters are determined by the validation data.

[2]Here, we use the Laplacian kernel $k(x_i, x_j) = \exp\left(-\|x_i - x_j\|/\sigma\right)$.

TABLE I
DATA DESCRIPTION: THE 20-NEWSGROUPS DATASETS

Task	# Fea.	# Doc.	Source Domain		Target Domain	
			# Pos.	# Neg.	# Pos.	# Neg.
Comp versus Sci (**C** versus **S**)	38 065	6000	1500	1500	1500	1500
Rec versus Talk (**R** versus **T**)	30 165	6000	1500	1500	1500	1500
Rec versus Sci (**R** versus **S**)	29 644	6000	1500	1500	1500	1500
Sci versus Talk (**S** versus **T**)	33 151	6000	1500	1500	1500	1500
Comp versus Rec (**C** versus **R**)	40 827	6000	1500	1500	1500	1500
Comp versus Talk (**C** versus **T**)	45 514	6000	1500	1500	1500	1500

TCA. As demonstrated in previous research [35], the manifold assumption holds on the WiFi data. Thus, the graph Laplacian term in SSTCA can effectively exploit label information from the labeled data to the unlabeled data across domains.

3) Comparison with Nonadaptive Methods: In this experiment, we compare TCA and SSTCA with learning-based localization models that do not perform domain adaptation, including RLSR, LapRLSR, and KPCA. The dimensionalities of the latent spaces for KPCA, TCA and SSTCA are fixed at 15. These values are determined based on the first experiment in Section V-B.2.

Fig. 3(b) shows the performance when the number of unlabeled patterns in $\mathcal{D}_T^u$ varies. As can be seen, even with only a few unlabeled data in the target domain, TCA and SSTCA can perform well for domain adaptation.

4) Comparison with Domain Adaptation Methods: In this section, we compare TCA and SSTCA with some state-of-the-art domain adaptation methods, including KMM, KLIEP, SCL, and SSA. We fix the dimensionalities of the latent space in TCA and SSTCA at 15, while we fix the dimensionalities of the latent space in SSA and TCAReduced at 50. For training, all the source domain data are used and varying amounts of the target domain data are sampled as $|D_T^u|$.

Results are shown in Fig. 3(c). As can be seen, domain adaptation methods that are based on feature extraction (including SCL, TCA, and SSTCA) perform much better than instance reweighting methods (including KMM and KLIEP). This is again because the WiFi data are highly noisy, and so matching distributions directly based on the noisy observations may not be useful. Indeed, SCL may suffer from the bad choice of pivot features due to the noisy observations. On the other hand, TCA and SSTCA match distributions in the latent space, where the WiFi data have been implicitly denoised.

5) Comparison with MMDE: In this section, we compare TCA and SSTCA with MMDE in the transductive setting. The latent space is learned from $\mathcal{D}_S$ and a subset of the unlabeled target domain data sampled from $\mathcal{D}_T^u$. The performance is then measured on $\mathcal{D}_T^u$.

Fig. 4(a) shows the results for different dimensionalities of the latent space with $|\mathcal{D}_T^u| = 400$, while Fig. 4(b) shows the results for different amounts of unlabeled target domain data, with the dimensionalities of MMDE, TCA, and SSTCA fixed at 15. As can be seen, MMDE outperforms TCA and SSTCA. This may be due to the limitation that the kernel matrix used in TCA/SSTCA is parametric. However, as mentioned in Section IV-C, MMDE is computationally expensive because it involves an SDP. This is confirmed in the training time comparison in Fig. 4(c). In practice, TCA or SSTCA may be a better choice than MMDE.

6) Sensitivity to Parameters: In this section, we investigate the effects of the parameters on the regression performance. These include the kernel width σ in the Laplacian kernel, tradeoff parameter μ, and for SSTCA, the two additional parameters γ and λ. The out-of-sample evaluation setting is used. All the source domain data are used, and we sample 2328 samples from the target domain data to form $\mathcal{D}_T^o$, and another 400 samples to form $\mathcal{D}_T^u$. The dimensionalities of the latent spaces in TCA and SSTCA are fixed at 15. As can be seen from Fig. 5, both TCA and SSTCA are insensitive to various settings of the parameters.

C. Cross-Domain Text Classification

1) Experimental Setup: In this section, we perform cross-domain text classification experiments on a preprocessed dataset of the 20-Newsgroups [36]. In this experiment, we follow the preprocessing strategy in [37] to create six datasets from this collection. For each dataset, two top categories are chosen, one as positive and the other as negative. We then split the data based on subcategories. Different subcategories are considered as different domains, and the binary classification task is defined as top category classification. This splitting strategy ensures that the domains of labeled and unlabeled data are related, since they are under the same top categories. Besides, the domains are also ensured to be different, since they are drawn from different subcategories. The six data sets created are "comp versus sci," "rec versus talk," "rec versus sci," "sci versus talk," "comp versus rec," and "comp versus talk" (Table I).

From each of these six datasets, we randomly sample 40% of the documents from the source domain as $\mathcal{D}_S$, and sample 40% from the target domain to form the unlabeled subset $\mathcal{D}_T^u$, and the remaining 60% in the target domain to form the out-of-sample subset $\mathcal{D}_T^o$. Hence, in each cross-domain classification task, $|\mathcal{D}_S| = 1200$, $|\mathcal{D}_T^u| = 1200$, and $|\mathcal{D}_T^o| = 1800$. We run 10 repetitions and report the average results. All experiments are performed in the out-of-sample setting. The evaluation criterion is the classification accuracy.

Similar to Section V-B, we perform a series of experiments to compare TCA and SSTCA with the following methods. 1) Linear support vector machine (SVM) in the original input space. 2) KPCA. A linear SVM is then trained in the latent space. 3) Three domain adaptation methods: KMM, KLIEP, and SCL. Again, a linear SVM is used as the classifier in the latent space. We experiment with the radial basis

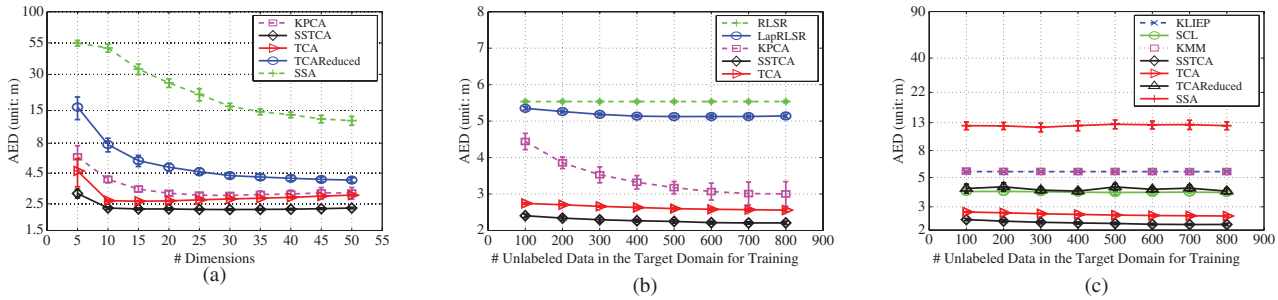

Fig. 3. Comparison of TCA, SSTCA and the various baseline methods in the inductive setting on the WiFi data. (a) Results on dimensionality reduction methods. (b) Comparison with nonadaptive methods. (c) Comparison with domain adaptation methods.

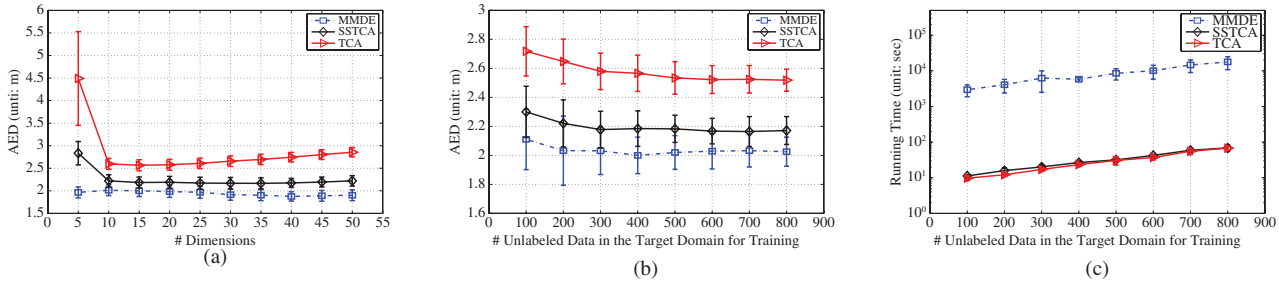

Fig. 4. Comparison with MMDE in the transductive setting on the WiFi data. (a) Varying the dimensionality of the latent space. (b) Varying the number of unlabeled data. (c) Comparison on training time.

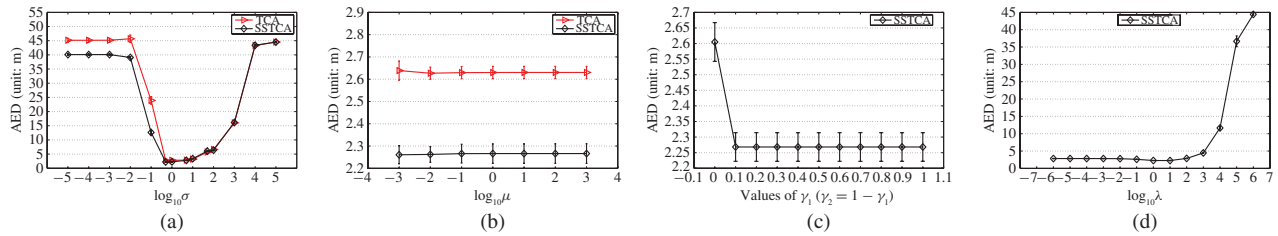

Fig. 5. Sensitivity analysis of the TCA/SSTCA parameters on the WiFi data. (a) Varying σ of the Laplacian kernel. (b) Varying μ. (c) Varying γ in SSTCA. (d) varying λ in SSTCA.

function (RBF), Laplacian, and linear kernels for feature extraction or reweighting in KPCA, KMM, TCA and SSTCA. Note that we do not compare with SSA because it results in "out of memory" on computing the covariance matrices. For SSTCA, kernel k_{yy} in (9) is the linear kernel. The μ parameters in TCA and SSTCA are set to 1, and the λ parameter in SSTCA is set to 0.0001.

2) Results: Results are shown in Table II. As can be seen, we can obtain a similar conclusion as in Section V-B. Overall, feature extraction methods outperform instance-reweighting methods. In addition, on tasks such as "**R** versus **T**," "**C** versus **T**," "**C** versus **S**," and "**C** versus **R**," the performance of PCA is comparable to that of linear TCA. However, on tasks such as "**R** versus **S**" and "**S** versus **T**," linear TCA performs much better than PCA. This agrees with our motivation and the previous conclusion on the WiFi experiments, namely that mapping data from different domains to a latent space spanned by the principal components may not work well, as PCA cannot guarantee a reduction in distance of the two domain distributions. In general, one may notice two main differences between the results on the WiFi data and those on the text data. First, the linear kernel performs better than the RBF and Laplacian kernels here. This agrees with the well-known observation that the linear kernel is often adequate for high-dimensional text data. Moreover, TCA performs better than SSTCA on the text data. This may be because the manifold assumption is weaker in the text domain than in the WiFi domain.

We also test how the various parameters in TCA and SSTCA affect the classification performance. For this test, we use linear kernels for both the inputs and outputs, and the dimensionalities of the latent spaces are fixed at 10. Thus, the remaining free parameters are μ for TCA, and μ, γ and λ for SSTCA. Due to the limitation of space, we do not show the detailed results of these experiments. Similar to the results on the WiFi data, both TCA and SSTCA are insensitive to the setting of μ. In addition, there is a wide range of λ for which the performance of SSTCA is quite stable. However, different from the WiFi results in Fig. 5(d) where SSTCA performs well when $\lambda \leq 10^2$, on the text datasets SSTCA performs well only when λ is very small ($\lambda \leq 10^{-4}$). This indicates that manifold regularization may not be useful on this text data.

TABLE II
CLASSIFICATION ACCURACIES (%) OF THE VARIOUS METHODS (THE NUMBER INSIDE PARENTHESES IS THE STANDARD DEVIATION)

method		#dim	task					
			C versus **S**	**R** versus **T**	**R** versus **S**	**S** versus **T**	**C** versus **R**	**C** versus **T**
SVM		all	68.26 (1.23)	72.33 (2.32)	75.86 (1.55)	76.70 (1.05)	81.59 (1.36)	90.51 (0.70)
TCA	linear kernel	10	70.41 (6.84)	87.67 (2.12)	82.83 (3.07)	**84.51 (5.04)**	89.79 (2.54)	88.67 (3.01)
		20	69.04 (5.07)	81.52 (8.86)	83.86 (3.24)	82.23 (2.64)	89.73 (4.04)	92.03 (2.05)
		30	69.01 (2.39)	77.26 (15.81)	**84.44 (2.29)**	79.81 (4.20)	**91.81 (2.38)**	92.23 (2.41)
	Laplacian kernel	10	69.12 (13.27)	78.68 (8.23)	79.58 (8.12)	69.02 (5.31)	82.29 (2.57)	90.01 (1.02)
		20	69.37 (11.22)	79.57 (4.31)	78.71 (9.22)	74.71 (3.01)	85.59 (1.59)	92.68 (1.12)
		30	68.58 (11.21)	80.43 (4.64)	76.69 (9.52)	74.40 (2.49)	87.89 (2.22)	92.63 (0.84)
	RBF kernel	10	74.88 (3.51)	82.51 (7.65)	78.34 (6.58)	81.65 (4.08)	82.69 (2.24)	89.15 (0.69)
		20	72.60 (5.60)	77.47 (2.62)	78.09 (6.88)	79.54 (1.89)	83.51 (3.32)	90.77 (0.83)
		30	71.64 (5.49)	77.62 (3.75)	80.11 (7.73)	79.50 (1.91)	83.71 (2.27)	91.58 (0.64)
SSTCA	linear kernel	10	68.64 (3.00)	75.11 (11.93)	81.46 (3.59)	73.75 (7.55)	85.99 (3.22)	91.38 (2.22)
		20	64.28 (3.48)	60.69 (14.87)	77.45 (5.30)	78.19 (4.17)	86.71 (3.36)	91.81 (2.13)
		30	65.08 (3.12)	66.30 (16.74)	77.98 (4.19)	72.79 (5.75)	85.81 (3.23)	**93.38 (2.02)**
	Laplacian kernel	10	**75.29 (3.92)**	79.84 (5.03)	72.70 (10.69)	73.10 (2.10)	85.26 (2.25)	91.72 (0.64)
		20	71.99 (4.73)	81.77 (3.44)	72.99 (9.99)	74.94 (2.28)	84.28 (1.27)	92.47 (0.74)
		30	69.71 (4.99)	82.09 (4.42)	72.34 (10.82)	74.67 (1.79)	85.30 (1.80)	92.73 (0.76)
	RBF kernel	10	73.76 (3.25)	74.50 (7.85)	78.51 (7.50)	77.61 (1.49)	83.09 (.0287)	90.35 (1.18)
		20	70.87 (7.51)	75.49 (6.67)	79.28 (7.20)	79.46 (1.27)	80.02 (.0287)	90.62 (0.83)
		30	70.16 (5.98)	77.03 (5.56)	79.06 (7.60)	79.88 (1.52)	81.30 (.0287)	90.21 (0.96)
KPCA	linear kernel	10	68.66 (6.59)	**88.26 (5.85)**	68.59 (10.00)	81.42 (6.67)	87.33 (3.56)	91.24 (1.84)
		20	69.18 (6.27)	82.59 (7.07)	71.46 (7.41)	80.22 (3.81)	89.49 (3.34)	93.44 (1.92)
		30	70.55 (2.81)	80.94 (11.63)	78.90 (8.33)	77.92 (4.32)	91.36 (1.51)	**93.66 (1.81)**
	Laplacian kernel	10	44.43 (8.01)	81.52 (9.00)	54.42 (7.33)	80.37 (.0252)	58.87 (4.97)	58.47 (2.35)
		20	49.08 (10.46)	55.67 (6.35)	50.42 (1.01)	72.67 (.0252)	75.71 (6.83)	73.94 (3.75)
		30	45.24 (8.17)	63.13 (7.76)	50.43 (1.03)	69.36 (.0252)	75.07 (10.64)	74.18 (4.24)
	RBF kernel	10	53.82 (6.23)	78.50 (4.23)	51.64 (2.11)	79.92 (.0252)	57.84 (3.74)	56.82 (2.03)
		20	47.66 (8.19)	60.94 (10.97)	50.49 (1.00)	79.37 (.0252)	67.73 (5.53)	62.36 (3.84)
		30	47.82 (8.37)	69.13 (9.66)	51.86 (3.82)	72.31 (.0252)	67.66 (4.48)	64.76 (5.14)
SCL		all+50	68.29 (1.22)	72.38 (2.36)	75.87 (1.48)	76.73 (1.00)	81.60 (1.35)	90.61 (0.64)
KMM	linear kernel	all	69.81 (1.27)	72.86 (1.53)	75.29 (1.85)	76.38 (1.32)	78.17 (1.29)	88.06 (1.33)
	Laplacian kernel	all	69.64 (1.27)	73.10 (1.67)	76.62 (1.23)	75.83 (1.27)	77.81 (1.21)	85.92 (0.70)
	RBF kernel	all	69.65 (1.24)	73.07 (1.48)	76.63 (1.14)	76.43 (1.17)	77.28 (1.18)	84.30 (1.00)
KLIEP		all	68.55 (1.36)	72.23 (1.20)	75.53 (1.17)	75.11 (0.81)	77.94 (1.09)	85.12 (0.99)

In this case, using *unsupervised* TCA for domain adaptation may be a better choice than using SSTCA.

VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel feature extraction method, TCA, for domain adaptation. It learns a set of transfer components in a RKHS such that when projecting domain data onto the latent space spanned by the transfer components, the distance between domains can be reduced. In order to capture the label dependence in transfer components learning, we further proposed a semisupervised feature extraction method, i.e., SSTCA, which can reduce the distance in data distributions between domains and maximize label dependence in a latent space simultaneously. Compared to the previous domain adaptation methods, TCA and SSTCA match distributions in a denoised latent space instead of the original feature space. Experiments on toy datasets and two real-world applications verify the efficiency and effectiveness of the proposed TCA and SSTCA.

In the future, we plan to continue our work by pursuing several avenues. First, we plan to adaptively estimate the number of transfer components in TCA and SSTCA. Second, in order to speed up kernel learning for domain adaptation, TCA and SSTCA propose to use parametric kernels for the MMD measure, we plan to develop an efficient algorithm for kernel choice in TCA and SSTCA. Moreover, we also plan to extend TCA and SSTCA to a multidomain setting.

ACKNOWLEDGMENT

The authors would like to thank P. von Bünau for providing the code of stationary subspace analysis.

REFERENCES

[1] S.-H. Fang and T.-N. Lin, "Indoor location system based on discriminant-adaptive neural network in IEEE 802.11 environments," *IEEE Trans. Neural Netw.*, vol. 19, no. 11, pp. 1973–1978, Nov. 2008.
[2] Q. Yang, S. J. Pan, and V. W. Zheng, "Estimating location using Wi-Fi," *IEEE Intell. Syst.*, vol. 23, no. 1, pp. 8–13, Jan.–Feb. 2008.
[3] J. Ghosn and Y. Bengio, "Bias learning, knowledge sharing," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 748–765, Jul. 2003.
[4] S. Ozawa, A. Roy, and D. Roussinov, "A multitask learning model for online pattern recognition," *IEEE Trans. Neural Netw.*, vol. 20, no. 3, pp. 430–445, Mar. 2009.
[5] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[6] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press, 2007, pp. 137–144.
[7] L. Song, A. Smola, K. Borgwardt, and A. Gretton, "Colored maximum variance unfolding," in *Advances in Neural Information Processing Systems 20*. Cambridge, MA: MIT Press, 2008, pp. 1385–1392.
[8] H. Daumé, III, "Frustratingly easy domain adaptation," in *Proc. 45th Ann. Meeting Assoc. Comput. Linguistics*, Prague, Czech Republic, Jun. 2007, pp. 256–263.
[9] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proc. Conf. Empirical Methods Natural Lang.*, Sydney, Australia, Jul. 2006, pp. 120–128.
[10] S. J. Pan, J. T. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *Proc. 23rd AAAI Conf. Artif. Intell.*, Chicago, IL, Jul. 2008, pp. 677–682.
[11] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *J. Mach. Learn. Res.*, vol. 6, pp. 1817–1853, Nov. 2005.

[12] P. von Bünau, F. C. Meinecke, F. C. Kirúly, and K. R. Müller, "Finding stationary subspaces in multivariate time series," *Phys. Rev. Lett.*, vol. 103, no. 21, pp. 214101-1–214101-4, Sep. 2009.

[13] J. Suykens, "Data visualization and dimensionality reduction using kernel maps with a reference point," *IEEE Trans. Neural Netw.*, vol. 19, no. 9, pp. 1501–1517, Sep. 2008.

[14] J. Huang, A. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf, "Correcting sample selection bias by unlabeled data," in *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press, 2007, pp. 601–608.

[15] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Advances in Neural Information Processing Systems 20*. Cambridge, MA: MIT Press, 2008, pp. 1433–1440.

[16] S. Bickel, M. Brückner, and T. Scheffer, "Discriminative learning under covariate shift," *J. Mach. Learn. Res.*, vol. 10, pp. 2137–2155, Sep. 2009.

[17] T. Kanamori, S. Hido, and M. Sugiyama, "A least-squares approach to direct importance estimation," *J. Mach. Learn. Res.*, vol. 10, pp. 1391–1445, Jul. 2009.

[18] M. Sugiyama, M. Kawanabe, and P. L. Chui, "Dimensionality reduction for density ratio estimation in high-dimensional spaces," *Neural Netw.*, vol. 23, no. 1, pp. 44–59, Jan. 2010.

[19] T. Suzuki and M. Sugiyama, "Estimating squared-loss mutual information for independent component analysis," in *Proc. 8th Int. Conf. Independent Compon. Anal. Signal Separation*, Paraty, Brazil, Mar. 2009, pp. 130–137.

[20] S. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori, "Inlier-based outlier detection via direct density ratio estimation," in *Proc. 8th IEEE Int. Conf. Data Mining*, Pisa, Italy, Dec. 2008, pp. 223–232.

[21] Y. Kawahara and M. Sugiyama, "Change-point detection in time-series data by direct density-ratio estimation," in *Proc. SIAM Int. Conf. Data Mining*, Sparks, NV, Apr. 2009, pp. 389–400.

[22] A. J. Smola, A. Gretton, L. Song, and B. Schölkopf, "A Hilbert space embedding for distributions," in *Proc. 18th Int. Conf. Algorithmic Learn. Theory*, Sendai, Japan, Oct. 2007, pp. 13–31.

[23] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, "A kernel method for the two-sample problem," in *Proc. Conf. Neural Inf. Process. Syst. 19*. Cambridge, MA, 2007, pp. 513–520.

[24] A. Gretton, O. Bousquet, A. J. Smola, and B. Schölkopf, "Measuring statistical dependence with Hilbert-Schmidt norms," in *Proc. 18th Int. Conf. Algorithmic Learn. Theory*, Singapore, Oct. 2005, pp. 63–77.

[25] I. Steinwart, "On the influence of the kernel on the consistency of support vector machines," *J. Mach. Learn. Res.*, vol. 2, pp. 67–93, Nov. 2001.

[26] K. Q. Weinberger, F. Sha, and L. K. Saul, "Learning a kernel matrix for nonlinear dimensionality reduction," in *Proc. 21st Int. Conf. Mach. Learn.*, Banff, AB, Canada, Jul. 2004, pp. 839–846.

[27] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.

[28] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–201, Mar. 2001.

[29] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006.

[30] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, "On kernel-target alignment," in *Advances in Neural Information Processing Systems 14*. Cambridge, MA: MIT Press, 2002, pp. 367–373.

[31] H. Xiong, M. Swamy, and M. Ahmad, "Optimizing the kernel in the empirical feature space," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 460–474, Mar. 2005.

[32] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Nov. 2006.

[33] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. Philadelphia, PA: SIAM, 1994.

[34] D. C. Sorensen, "Implicitly restarted Arnoldi/Lanczos methods for large scale eigenvalue calculations," Dept. Comput. Appl. Math., Rice Univ., Houston, TX, Tech. Rep. TR-96-40, 1996.

[35] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," in *Proc. 21st Int. Joint Conf. Artif. Intell.*, Pasadena, CA, Jul. 2009, pp. 1187–1192.

[36] K. Lang, "Newsweeder: Learning to filter netnews," in *Proc. 12th Int. Conf. Mach. Learn.*, San Mateo, CA, 1995, pp. 331–339.

[37] W. Dai, G. Xue, Q. Yang, and Y. Yu, "Co-clustering based classification for out-of-domain documents," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, San Jose, CA, Aug. 2007, pp. 210–219.

Sinno Jialin Pan received the B.S. and M.S. degrees in applied mathematics from Sun Yat-sen University, Guangzhou, China, in 2003 and 2005, respectively, and the Ph.D. degree in computer science and engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2010.

He is currently a Research Fellow at the Institute for Infocomm Research, Singapore. His current research interests include transfer learning and its applications in pervasive computing, bioinformatics, and web mining such as opinion mining, text mining, and information extraction.

Ivor W. Tsang received the Ph.D. degree in computer science from the Hong Kong University of Science and Technology, Hong Kong, in 2007.

He is currently an Assistant Professor with the School of Computer Engineering, Nanyang Technological University (NTU), Singapore. He is also the Deputy Director of the Center for Computational Intelligence, NTU.

Dr. Tsang received the IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award for 2004 in 2006, and the second-class Prize of the National Natural Science Award 2008, China, in 2009. He was awarded the Microsoft Fellowship in 2005 and the Best Paper Award from the IEEE Hong Kong Chapter of the Signal Processing Postgraduate Forum in 2006. His work was also awarded the Best Student Paper Prize at CVPR'10.

James T. Kwok received the Ph.D. degree in computer science from the Hong Kong University of Science and Technology, Hong Kong, in 1996.

He was with the Department of Computer Science, Hong Kong Baptist University, Hong Kong, as an Assistant Professor. Since 2000, he has been an Associate Professor in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His current research interests include kernel methods, machine learning, pattern recognition, and artificial neural networks.

Dr. Kwok received the IEEE Outstanding Paper Award in 2006, and the second-class Prize of the National Natural Science Award 2008, China, in 2009. He is currently an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS and the *Neurocomputing Journal*.

Qiang Yang (F'02) received the B.S. degree in astrophysics from Peking University, Beijing, China, and the Ph.D. degree in computer science from the University of Maryland, College Park.

He is now a Professor in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong. His current research interests include data mining and artificial intelligence.

Prof. Yang was elected a Vice-Chair of the Association for Computing Machinery (ACM) special interest group on artificial intelligence in July 2010. He is the founding Editor-in-Chief of the *ACM Transactions on Intelligent Systems and Technology*. He has been a PC Co-Chair or General Co-Chair for a number of international conferences. He was an invited speaker at the International Joint Conference on Artificial Intelligence'09, ACL'09, ACML'09, and ADMA'08.