Semi-supervised Domain Adaptation on Manifolds

Li Cheng, *Senior Member, IEEE*, and Sinno Jialin Pan

***Abstract*— In real-life problems, the following semi-supervised domain adaptation scenario is often encountered: we have full access to some source data, which is usually very large; the target data distribution is under certain unknown transformation of the source data distribution; meanwhile, only a small fraction of the target instances come with labels. The goal is to learn a prediction model by incorporating information from the source domain that is able to generalize well on the target test instances. We consider an explicit form of transformation functions and especially linear transformations that maps examples from the source to the target domain, and we argue that by proper preprocessing of the data from both source and target domains, the feasible transformation functions can be characterized by a set of rotation matrices. This naturally leads to an optimization formulation under the special orthogonal group constraints. We present an iterative coordinate descent solver that is able to jointly learn the transformation as well as the model parameters, while the geodesic update ensures the manifold constraints are always satisfied. Our framework is sufficiently general to work with a variety of loss functions and prediction problems. Empirical evaluations on synthetic and real-world experiments demonstrate the competitive performance of our method with respect to the state-of-the-art.**

***Index Terms*— Domain adaptation, semi-supervised learning, transfer learning.**

I. Introduction

SUPERVISED learning methods have been shown to perform well on many real-world applications, such as computer vision, natural language processing (NLP), to name a few. However, these methods usually assume the test data are drawn from the same distribution as the training data, and often fail to make satisfactory predictions on novel domains. Domain adaptation algorithms, on the other hand, aims to address such learning scenarios where a very few or even no labeled data are available in a related target domain. This is often realized by having access to a sufficient amount of labeled training data from source domains, even when these domains have different data distributions. The methods have been empirically [1]–[3] as well as theoretically [4], [5] studied for these cross-domain learning problems.

In domain adaptation [6], a key research challenge is how to learn a proper feature representation for both source and target domains such that training data from both domains can be used to address the prediction task in the target domain, including, for example, cross-domain object recognition where the source and target domains are from different image modalities, or cross-domain WiFi localization where the source and target domains refer to data collected from different time periods. Blitzer *et al.* [7] proposed structural correspondence learning (SCL) to learn common features by inducing correspondences among features from different domains. SCL is motivated by a multitask learning algorithm [8] to select pivot features between the source and target domains, to construct pseudotasks for building correspondences among the features. Daumé [1] proposed a feature augmentation (FA) method for domain adaptation in NLP, which augments the source domain feature space using features from a few labeled data in the target domain. Pan *et al.* [3] proposed a dimensionality reduction method to learn a latent space underlying different domains, by reducing distance of data distributions between the domains while still preserving data variance and local structure. Most of the previous methods can be either referred to as unsupervised domain adaptation approaches [3], [7], which do not take label information into consideration when learning the feature representation, or as supervised domain adaptation approaches [1], which only use labeled data from the source and target domains.

In this paper, we consider the relatively less-explored paradigm of semi-supervised domain adaptation, which is to leverage both labeled and unlabeled data in the target domain, as well as labeled data in the source domain. Daumé *et al.* [9] extended FA in a semi-supervised manner (semi-FA), which builds on the notion of augmented feature space proposed by FA and harnesses unlabeled data in the target domain to further assist knowledge transfer from the source domain to the target domain. Donahue *et al.* [10] advocated the usage of known constraints from unlabeled instances to facilitate object detection and classification tasks with additional source domain data. A semi-supervised feature extraction method, semi-supervised transfer component analysis or SSTCA, was developed in [11] where extensive empirical results on text classification and WiFi localization are shown. Nevertheless, both semiFA and SSTCA are two-step approaches to domain adaptation, which first learn the feature representation across the domains, and models are subsequently trained based on this feature to make prediction on the target domain. The work of [12] considers transferring learning across the multiple tasks under a semi-supervised Bayesian scenario. In addition, [13] provides a systematic coverage of semi-supervised learning.

We propose a novel semi-supervised domain adaptation framework that explicitly considers the mapping or transformation function from the source to the target feature space. While being capable of working with nonlinear transformation,

Manuscript received July 31, 2013; accepted February 22, 2014. This work was supported in part by A*STAR JCO and in part by A*STAR IAF.

L. Cheng is with the Bioinformatics Institute, A*STAR, Singapore 138671 (e-mail: chengli@bii.a-star.edu.sg).

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

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/TNNLS.2014.2308325

in this paper, we will mainly focus on linear transformation. More specifically, we consider the scenarios where after proper preprocessing (including mean subtraction and normalization by scaling), the set of feasible transformations are closed under rotation, i.e., the orthogonal group constraints. We then develop a manifold-based gradient descent algorithm that is able to jointly learn the model parameters and the transformation, and demonstrate its capacity of working with a variety of loss functions and prediction (i.e., classification and regression) problems.

Related to our method are the metric learning approaches for domain adaptation for the tasks of object recognition [14], [15]. The main idea is to learn a transformation from source to target using learning the metrics with explicit cross-domain constraints. Once the transformation is learned, classifiers are trained on the source domain data, then used to make predictions on new instances from the transformed target domain. Different from this line of approaches where only labeled data from target domain are considered, our framework exploits both labeled and unlabeled data in the target domain, to learn the transformation from source to target. Furthermore, our framework is able to learn the transformation and the model parameters simultaneously instead of learning them separately. It is worth pointing out that the geodesic update adopted by our algorithm guarantees the manifold constraints are always satisfied, while it is clearly not the case for the metric learning methods where there are no specific constraints on the set of feasible transformations. Domain adaptation machine (DAM) [16] is another related work, which is a semi-supervised extension of SVMs for domain adaptation. In DAM, the knowledge carried by the source domain is implicitly encoded in a data-dependent regularization term instead of explicit transformation, which may not be able to fully explore the source domain knowledge during the adaptation process. Furthermore, DAM is customized for SVMs, while our method works with a range of loss functions.

II. Preparation

A. Graph Laplacian

A graph G consists of an ordered and finite set of n vertices (i.e., instances) X, and a finite set of edges E each connecting pairs of these instances. A vertex x_i is said to be a neighbor of another vertex x_j if they are connected by an edge. G is said to be undirected if $(x_i, x_j) \in E \iff (x_j, x_i) \in E$ for all edges. The weighted adjacency matrix of G is an $n \times n$ real matrix ψ with $\psi_{ij} \in (0, \infty)$ if $(x_i, x_j) \in E$, and 0 otherwise. Let D, the degree matrix, be an $n \times n$ diagonal matrix with entries $D_{ii} = \sum_j \psi_{ij}$. The graph Laplacian is the matrix $L = D - \psi$ while the normalized graph Laplacian $\Delta_G := D^{-1/2}\mathrm{LD}^{-1/2}$.

In what follows, we briefly recall the related notions of matrix manifold and matrix Lie group. Motivated readers can refer to [17] and [18, Ch. 20] for further details.

B. Special Orthogonal Group

Define the set of $d \times d$ orthonormal matrices as $\mathrm{SO}(d) := \{\Phi \in \mathcal{R}^{d\times d} : \Phi\Phi^\top = \Phi^\top\Phi = I_d,\ \det(\Phi) = 1\}$, where I_d is an identity matrix of size $d \times d$. It is a matrix Lie group, being a matrix group as well as a differentiable manifold that can be locally approximated by a set of Euclidean spaces. Consider an arbitrary point $\Phi \in \mathrm{SO}(d)$. To perform differential calculus, define the tangent space at Φ as $T_\Phi\mathrm{SO}(d)$, which is a subset of $d \times d$ matrices. It is easy to check $\Phi^\top\rho = -\rho^\top\Phi$ for any tangent vector $\rho \in T_\Phi\mathrm{SO}(d)$. That is, $\Phi^\top\rho$ is skew-symmetric.[1] Define its metric as an inner product $\langle\rho_1, \rho_2\rangle = \mathrm{trace}(\rho_1^\top\rho_2)$ for any $\rho_1, \rho_2 \in T_\Phi\mathrm{SO}(d)$. With the metric definition we have a Riemannian manifold. Now, given a differentiable function l mapping from $\Phi \in \mathrm{SO}(d)$ to $\mathcal{R}$, and $\nabla_\Phi l$ its gradient in the local Euclidean space (also a matrix of $d \times d$), its Riemannian gradient $\tilde{\nabla}_\Phi l \in \mathrm{TSO}(d)$ on the embedded manifold is computed as

$$\tilde{\nabla}_\Phi l := \nabla_\Phi l - \Phi\nabla_\Phi l^\top\Phi. \tag{1}$$

In a nutshell, it projects the Euclidean gradient onto the current tangent space. Denote a skew-symmetrization operator $\mathrm{skew}(A) = A - A^\top$ for any square matrix A. It is easy to check that $\Phi^\top\tilde{\nabla}_\Phi l = \mathrm{skew}(\Phi^\top\nabla_\Phi l)$ always holds true.

C. Exponential Map, Retraction

Intuitively, retractions generalize the notion of moving in the direction of a vector in Euclidean space to manifolds. An ideal retraction is the exponential map. In our context, the exponential map at point Φ, $R_\Phi(\rho)$, maps a tangent vector $\rho \in T_\Phi\mathrm{SO}(d)$ to a point in the manifold $\mathrm{SO}(d)$, as projecting along a geodesic curve started at Φ in the direction of ρ. In practice, usually a computationally less demanding retraction is used instead of the exact exponential map. Formally, a retraction on a manifold $\mathrm{SO}(d)$ at point Φ is a function $R_\Phi : T_\Phi\mathrm{SO}(d) \to \mathrm{SO}(d)$ that satisfies two properties [19]. 1) $R_\Phi(0) = \Phi$. 2) The geodesic curve defined by $\gamma_\Phi(t) := R_\Phi(t\rho)$ satisfies $\dot{\gamma}_\Phi(0) = \rho$. In general, a retraction can approximate the exponential map at least to its first-order Taylor expansion [19, Sec. IV].

III. Our Approach

During training phase, we have access to the source domain $\hat{X} = (\hat{x}_1, \ldots, \hat{x}_{\hat{n}})$ a set of instances, and $\hat{Y} = (\hat{y}_1, \ldots, \hat{y}_{\hat{n}})$ their corresponding labels. Assume, we are also presented with a related target domain that contains a set of partially labeled examples: $X = (x_1, \ldots, x_m, \ldots, x_n)$ and the labels $Y = (y_1, \ldots, y_m)$, where $m \ll n$. The labels from both domains usually share the same output space, as $\hat{y}, y \in \mathcal{Y}$, meanwhile the instances from both domains might reside in different spaces. To simplify the matter, we start by assuming $\hat{X}$ and X reside in spaces of the same dimensions, and they are related under a linear transformation, denoted as Φ. In other words, there exists a mapping function $\hat{\Phi}$ such that for any $\hat{x} \in \hat{X}$, the pair $(\hat{\Phi}(\hat{x}), \hat{y})$ will be a proper example in the target domain, with $\hat{\Phi}(\hat{x}) := \Phi\hat{x}$. A preprocessing step is applied to the input instances, which involves a simple translation and scaling

[1]In particular, for identity $I_d \in \mathrm{SO}(d)$, its tangent space spans the set of $d \times d$ skew-symmetric matrices.

process to prepare both source and target examples into a zero-mean unit hypersphere for both domains, respectively. After this preprocessing step, we argue that the transformation Φ now belongs to the set of nonsingular rotational matrices that satisfies orthonormal constraints, $\Phi\Phi^\top = I,\ \Phi^\top\Phi = I$. As such an orthogonal matrix has determinant either 1 or -1, we further restrict to the subgroup containing identity I (i.e., having determinant 1), which is the special orthogonal group.

Now, let us revisit the general setting where the two domains are of different dimensional and are related with a nonlinear transformation, which can still be dealt with by our approach under a polynomial feature space expansion: for example, each source instance x can be lifted to a high $\hat{d}$-dimensional vector space by considering e.g., monomial basis $x^{(i)}, x^{(i)}x^{(i)}, x^{(i)}x^{(j)}, \ldots$, up to order D. Note $x^{(i)}$ and $x^{(j)}$ are used to refer to the ith and jth dimension of an instance x, respectively. This is followed by a linear map from $\hat{d}$- to d-dimensional. It has been shown in [17] that in fact this defines a curved subspace known as Stiefel manifold. The capacity of our approach to work with the more general setting is illustrated later in empirical simulations (e.g., Fig. 4), where $\hat{d} \neq d$. In what follows, we will still mainly focus on the simpler scenario, i.e., $\hat{X}$ and X are of the same dimensions, and are related under a linear transformation, which is a reasonable assumption in many applications, and especially so in computer vision tasks including object recognition and detection.

Now, for an instance-label pair (x, y), consider a loss function $l(f_w(x), y)$ endowed with a linear form $f_w(x) = w^\top x$, and a regularizer $\Omega(w)$ of the model parameter $w \in \mathcal{R}^d$. We propose to formulate the induced optimization problem of domain adaption as

$$\min_{\Phi,w} \Omega(w) + \eta_1 \sum_{j=1}^{m} l\big(f_w(x_j), y_j\big) + \eta_2 \sum_{i=1}^{\hat{n}} l\big(f_w(\Phi\hat{x}_i), \hat{y}_i\big) \tag{2}$$

subjected to orthonormal constraints $\Phi \in SO(d)$, that is

$$\Phi\Phi^\top = I, \quad \Phi^\top\Phi = I, \quad \det(\Phi) = 1. \tag{3}$$

In other words, the set of feasible Φ matrices forms the special orthogonal group, $SO(d)$.

A. Graph-Based Semi-supervised Domain Adaptation

Graph-based semi-supervised methods construct a problem graph, G, whose nodes are the training instances (both labeled and unlabeled) of the target domain, and edges encode nearest neighbor relationships. Often times, the edges are weighted by a kernel function to reflect the similarity between neighboring examples. The semi-supervised learning problem can indeed be posed as that of estimating a smooth function that respects neighborhood relations on the graph. Let X denote the target instance matrix of size $d \times n$, containing both labeled and unlabeled instances. Following [20] and others, the inductive Laplacian graph regularizer $||w||_G^2$ is defined as

$$\|w\|_G^2 := \boldsymbol{f}_w^\top \Delta_G \boldsymbol{f}_w = w^\top \big(X\Delta_G X^\top\big) w \tag{4}$$

where $\boldsymbol{f}_w$ denotes the vector $\big[f_w(x_1), \ldots, f_w(x_m), \ldots f_w(x_n)\big]^\top$, $\Delta_G \in \mathcal{R}^{n\times n}$ is the normalized Laplacian of G. To incorporate unlabeled target training examples into (2), we consider minimize the following regularized risk:

$$\min_{\Phi,w} \ \Omega(w) + \eta_1 \sum_{j=1}^{m} l\big(f_w(x_j), y_j\big) + \eta_2 \sum_{i=1}^{\hat{n}} l\big(f_w(\Phi\hat{x}_i), \hat{y}_i\big) + \eta_3 \|w\|_G^2 \tag{5}$$

subjected to proper constraints of (3).

B. Our Block Coordinate Descent Algorithm Using Armijo Rule

The problem of (5) is nonconvex, which can nevertheless be solved by considering a block coordinate descent algorithm (e.g., [21]) that iteratively minimizes over Φ and w. For this purpose, denote the $w-$related objective function

$$l_1(w) := \Omega(w) + \eta_1 \sum_{j=1}^{m} l(f_w(x_j), y_j) + \eta_2 \sum_{i=1}^{\hat{n}} l\big(f_w(\Phi\hat{x}_i), \hat{y}_i\big) + \eta_3 w^\top \big(X\Delta_G X^\top\big) w \tag{6}$$

and the $\Phi-$related objective, which is a loss function $l : \Phi \in \mathbb{V}_{d\times d} \to \mathcal{R}$ defined on $SO(d)$

$$l_2(\Phi) := \sum_{i=1}^{\hat{n}} l_2\big(f_w(\Phi\hat{x}_i), \hat{y}_i\big). \tag{7}$$

Denote $\nabla_\Phi l$, a $d \times d$ matrix, as the matrix derivative (i.e., Euclidean gradient) of the loss function $l_2(\Phi)$ of (7) in the local Euclidean space of Φ. Instead of explicitly minimize over the Lagrangian of the above constrained optimization problem of (7) subjected to (3), we consider a geodesic algorithm that performs minimization by moving along the underlying manifold.

Algorithm 1 presents a generic version of the proposed domain adaptation algorithm by gradient descent with self-tuning step sizes (i.e., λ_1 and λ_2) using the Armijo rule [19]. At each of the coordinate descent iterations, the model parameter w is updated by gradient descent following the usual optimization procedure in $\mathcal{R}^d$ [21]. To compute Φ, our algorithm involves two basic steps: 1) compute the Riemannian gradient by projecting the gradient onto the current tangent space and 2) bring the Riemannian gradient onto manifold along its geodesic curve by retraction.

For gradient projection, we have

$$\Phi^\top \tilde{\nabla}_\Phi l = \operatorname{skew}\big(\nabla_\Phi l\, \Phi^\top\big) \tag{8}$$

which can be directly obtained by right multiplying $\Phi^\top$ with both sides of (1). Consider simply the exponential map for retraction mapping, $R_\Phi(\rho) = \Phi\exp(-\Phi^\top\rho)$, we have a closed form update formula for the new Φ as

$$\begin{aligned}\Phi &:= R_\Phi\big(\eta_2\, \tilde{\nabla}_\Phi l\big)\\ &= \Phi\exp\big\{-\eta_2\, \operatorname{skew}\big(\nabla_\Phi l\, \Phi^\top\big)\big\}.\end{aligned} \tag{9}$$

Algorithm 1 Semi-supervised Domain Adaptation on Manifolds: Semi-DA

```
Require: Parameters $\eta_1$, $\eta_2$, $\eta_3$, and $\epsilon$
  Initialize step sizes $\lambda_1 \leftarrow 1$ and $\lambda_2 \leftarrow 1$
  repeat
    while $l_2(\Phi) - l_2\Big(R_\Phi\big(2\lambda_2\eta_2\,\tilde{\nabla}_\Phi l\big)\Big) \geq \lambda_2\eta_2\big\|\tilde{\nabla}_\Phi l\big\|_F^2$ do
      $\lambda_2 \leftarrow 2\lambda_2$
    end while
    while $l_2(\Phi) - l_2\Big(R_\Phi\big(\lambda_2\eta_2\,\tilde{\nabla}_\Phi l\big)\Big) < \frac{\lambda_2}{2}\eta_2\big\|\tilde{\nabla}_\Phi l\big\|_F^2$ do
      $\lambda_2 \leftarrow \frac{\lambda_2}{2}$
    end while
    Update $\Phi := R_\Phi\big(\eta_2\,\tilde{\nabla}_\Phi l\big)$
    while $l_1(w) - l_1\big(w - 2\lambda_1\eta_1\nabla_w l_1(w)\big) \geq \lambda_1\eta_1\big\|\nabla_w l_1(w)\big\|_F^2$ do
      $\lambda_1 \leftarrow 2\lambda_1$
    end while
    while $l_1(w) - l_1\big(w - \lambda_1\eta_1\nabla_w l_1(w)\big) < \frac{\lambda_1}{2}\eta_1\big\|\nabla_w l_1(w)\big\|_F^2$ do
      $\lambda_1 \leftarrow \frac{\lambda_1}{2}$
    end while
    Update $w := w - \lambda_1\eta_1\nabla_w l_1(w)$
  until $\big\|\tilde{\nabla}_\Phi l\big\|_F^2 < \epsilon$ & $\big\|\nabla_w l_1(w)\big\|_F^2 < \epsilon$
```

Note other forms of retractions are also possible [19, Ch. 4, p. 58]. Interestingly, for $\Phi \in SO(d)$, the exponential map can alternatively be interpreted as incorporating the Von Neumann divergence [22] term, $\Delta_F(\Phi, \Phi_t)$, into the minimization framework, thus the Φ-related optimization problem becomes

$$\min_{\Phi} \;\; \Delta_F(\Phi, \Phi_t) + \eta_2 \sum_{i=1}^{\hat{n}} l\big(f_w(\Phi\hat{x}_i), \hat{y}_i\big). \tag{10}$$

This gives rise to the iterative matrix exponentiated gradient update in [22], which is of the same form as the above exponential map update.

C. Loss Functions

The proposed approach can work with a variety of loss functions. Here, we focus only on the binary log loss, and relegate to the appendix details of other often-used loss functions, including, e.g., square, reverse prediction, and log losses. Specifically, consider a binary classification scenario with log loss and let $\Omega(w) := \|w\|_2^2/2$, we arrive at the following optimization problem:

$$\min_{w,\Phi} \;\; \frac{1}{2}\|w\|_2^2 + \eta_1 \sum_{j=1}^{m} \log\big(1 + e^{-y_j x_j^\top w}\big) + \eta_2 \sum_{i=1}^{\hat{n}} \log\big(1 + e^{-\hat{y}_i \hat{x}_i^\top \Phi^\top w}\big) + \eta_3\|w\|_G^2 \tag{11}$$

subjected to (3). Here, $l_2(\Phi) := \sum_{i=1}^{\hat{n}} \log\big(1 + e^{-\hat{y}_i\hat{x}_i^\top\Phi^\top w}\big)$, its gradient is

$$\nabla_\Phi l_2 = -\sum_{i=1}^{\hat{n}} \frac{\hat{y}_i \hat{x}_i w^\top}{1 + \exp\big(\hat{y}_i \hat{x}_i^\top \Phi w\big)} \tag{12}$$

and the corresponding manifold gradient can be subsequently computed by (1). On the other hand, we also have

$$l_1(w) := \frac{1}{2}\|w\|_2^2 + \eta_1 \sum_{j=1}^{m} \log\big(1 + e^{-y_j x_j^\top w}\big) + \eta_2 \sum_{i=1}^{\hat{n}} \log\big(1 + e^{-\hat{y}_i \hat{x}_i^\top \Phi^\top w}\big) + \eta_3\|w\|_G^2 \tag{13}$$

and its gradient

$$\nabla_w l_1 = -\eta_1 \sum_{j=1}^{n} \frac{y_j x_j}{1+\exp\big(y_j x_j^\top w\big)} - \eta_2 \sum_{i=1}^{\hat{n}} \frac{\hat{y}_i \Phi^\top \hat{x}_i}{1+\exp\big(\hat{y}_i \hat{x}_i^\top \Phi w\big)} + \big(\eta_3 X \Delta_G X^\top + I_d\big) w. \tag{14}$$

D. Convergence Analysis

We formulate the problem of semi-supervised domain adaptation as a constraint minimization problem with nonconvex objective function. In the following, we show that Algorithm 1 is guaranteed to converge to local optimal solutions.

Gradient descent methods using Armijo rule in Euclidean space $\mathcal{R}^n$ are shown (e.g., [23, Ch. 3]) to converge to local fixed points under mild conditions. It is also well known [19], [24] that on matrix manifolds they enjoy similar convergence properties as their analogs in Euclidian space. The proposed coordinate descent approach is guaranteed to decrease the objective function value monotonically as iteration increases: denote by $\mathcal{L}(\Phi, w)$ our objective function, at each iteration t we have $\mathcal{L}(\Phi^{t+1}, w^t) \leq \mathcal{L}(\Phi^t, w^t)$, as well as $\mathcal{L}(\Phi^{t+1}, w^{t+1}) \leq \mathcal{L}(\Phi^{t+1}, w^t)$. As $\mathcal{L}$ is compact, this ensures the limiting sequence produced by our algorithm will converge to a fixed point.

E. Computational Analysis

At each iteration, the main computational load is clearly about computing the retraction step. During the experiments, the exponential map is used directly, where the key is to compute the matrix exponential. In particular, we adopt the MATLAB function expm(), which is an approximation that uses the Pade method with scaling and squaring [25], and is known as one of the most efficient approximation methods [26]. The two major factors are the Pade order s and the scaling and squaring exponent e, where the overall complexity is $O((s+e)d^3)$ for both cases, with usually a large constant. Additionally, it usually takes $O(d^3)$ to compute both matrix multiplications and the Euclidean gradient for various loss functions.

F. Multisource

Assume, we have access to $\hat{D}$ source domains, and each domain indexed by $j \in \{1, \ldots, \hat{D}\}$ has $\hat{n}_j$ examples. It is straightforward to generalize our approach to multisource domain adaptation, by considering an extended version of (5) as

$$\min_{\{\Phi^{(j)}\},w} \quad \Omega(w) + \eta_1 \sum_{j=1}^{m} l\big(f_w(x_j), y_j\big) + \eta_2 \sum_{j=1}^{\hat{D}} \sum_{i=1}^{\hat{n}_j} l\Big(f_w\big(\Phi^{(j)}(\hat{x}_i^{(j)})\big), \hat{y}_i^{(j)}\Big) + \eta_3 \|w\|_G^2 \tag{15}$$

subjected to orthonormal constraints of matrices $\Phi^{(j)}$. Its solution is easily obtained as a variant of the aforementioned derivations.

G. Nonlinear Extension

So far, we have mainly considered linear transformation between the source and the target domains. As stated earlier, our approach can also work with the more general nonlinear scenario: for nonlinear transformation, each source instance x is first lifted to a high $\hat{d}$-dimensional vector space by considering e.g., monomial basis $x^{(i)}, x^{(i)}x^{(i)}, x^{(i)}x^{(j)}, \ldots$ up to order D, where $x^{(i)}$ and $x^{(j)}$ are used to refer to the ith and jth dimension of an instance x, respectively. This is followed by a linear map with dimensionality reduction such that only a subspace from source is used to transform to target vector space one-to-one, which implies $\hat{d} \geq d$. It has been shown in [17] that in fact this defines a curved subspace known as Stiefel manifold. In particular, when $\hat{d} = d$, it reduces to the special case of special orthogonal group.

IV. Empirical Studies

In this section, we first verify the motivations of our method on two synthetic data sets, and then apply the method to three real-world applications: cross-domain object recognition where the source and target domains are from different image modalities, cross-domain WiFi localization where the source and target domains refer to data collected from different time periods, and multisource sentiment classification where source and target are concerning Amazon online customer reviews from different product domains. In particular, the log loss function is used for the synthetic, the object recognition, and the sentiment classification experiments. To illustrate the applicability of our method working with other loss functions, we use the square loss in the WiFi localization problem. Throughout the experiments, we set ϵ to 0.01, λ_1 and λ_2 to 1.0. To exploit unlabeled training instances from target domain, G is constructed as a four-nearest neighbor graph.

A. Synthetic Data

For the first synthetic data set, a source domain consists of 400 instances drawn from two classes with equal prior probabilities (200 instances per class), where each class conditional probability assumes a separate multivariate Gaussian distribution. In the same manner, an additional set of 400 instances, which are referred to as a target domain, are sampled from these two class dependent distributions, then mapped to the target feature space under a random rotational transformation Φ as well as a random scaling and random translation transformations along each dimension. The difficulty levels of classification in each domain can be adjusted by moving distance between the two Gaussian means, while fixing the covariance matrices. Means (and covariance matrices) of the source class distributions are obtained by random vector (and matrix) generation, where specific care needs to be taken to ensure positive definite or PD covariance matrices. A random rotation transformation is then obtained by QR decomposition of a random PD matrix. The above data generation pipeline is executed to randomly generate 50 pairs of source and target domain data sets, where each pair corresponds to one cross-domain classification task. To simulate semi-supervised setting, the algorithm is presented with all 400 labeled source domain instances, together with only 40 labeled instances and 360 unlabeled instances from the target domain. The learned model is evaluated on the unlabeled target domain instances.

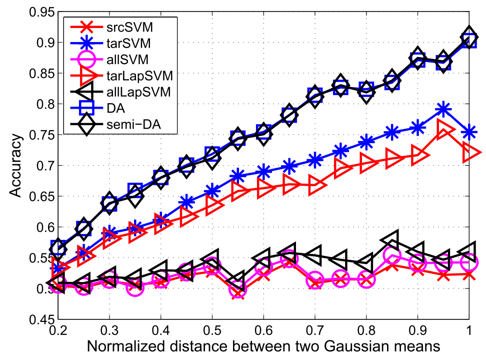
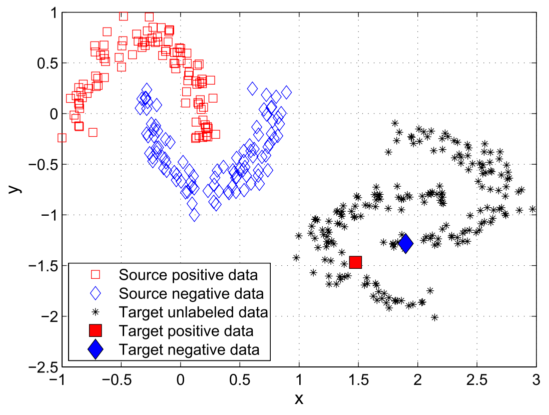


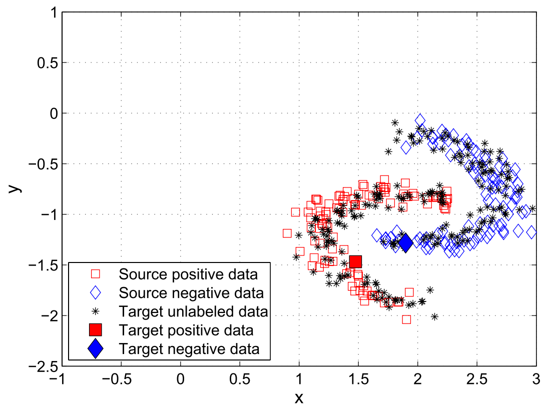
Fig. 1. Accuracy changes with the difficult levels of the tasks. Here, the difficult level is inversely related to the distance between the two classes' Gaussian means.

For comparison methods, as shown in Fig. 1, semi-DA refers to the proposed semi-supervised domain adaptation method using log loss; DA is a reduction of semi-DA by dropping the graph Laplacian regularizer, which can be referred to as a supervised domain adaptation method; srcSVM uses the binary SVM with source domain labeled data as input; tarSVM is the binary SVM working with labeled data from the target domain only; allSVM stands for the binary SVM with both source domain and target domain labeled data as input; tarLapSVM is the Laplacian SVM [20] incorporated with labeled and unlabeled data from the target domain; allLapSVM is the Laplacian SVM incorporated with labeled and unlabeled data from both the source and target domains. As expected, srcSVM and allSVM perform poorly by ignoring the transfer nature of the two domains, while by explicitly estimating the underlining rotation, our semi-DA and DA perform significantly better throughout different difficulty levels.

For the second synthetic data set, we first generate a normalized two-moon data set (zero-mean and identity covariance matrix) as a source domain, where each moon corresponds to a class, as shown on the left-hand side in Fig. 2. We then map the source two-moon data set to the target space under a random rotational transformation Φ as well as a random shifting transformation along each dimension. The mapped



(a)



(b)

Fig. 2. Example of 2-D two-moon data set. (a) Source and target two-moon data sets. (b) Source and target two-moon data sets after applying the learned transformation.

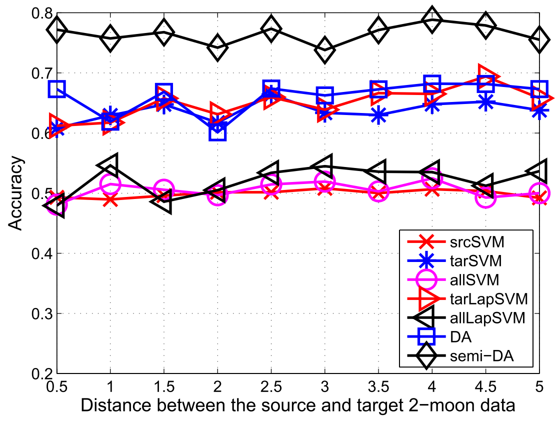


Fig. 3. Comparison results under varying distances between the source and target two-moon data.

two-moon data set is considered as a target domain, as shown on the right-hand side in Fig. 2. Similar to the semi-supervised setting of the first synthetic data set, for the target domain data, we randomly sample only one instance from each moon as labeled data, and consider the rest as unlabeled target domain data. The evaluation is performed on the unlabeled target domain data. Fig. 2 shows the result of semi-DA on the two-moon data sets shown in Fig. 2, where we first normalize the target domain data, and then perform semi-DA on the source and target domain data to learn a rotational transformation for the source domain data, and finally shift the transformed source domain data based on the originally estimated mean and covariance matrix of the target domain instances. As can be observed, the two moons of the source and target domain are aligned correctly, which is very helpful for classification in the target domain.

Based on the above data generation pipeline, we randomly generate 50 pairs of source and target domain data sets and their corresponding cross-domain classification tasks. The average results of all comparison methods are shown in Table I. From the table, we can observe that different from Fig. 1, the semi-supervised learning method tarLapSVM only with target domain data performs better than the supervised domain adaptation method DA. The reason may be that the manifold assumption holds in the target domain, exploiting the manifold structure from the unlabeled data may significantly boost the classification performance even though the number of labeled data is small. However, since the manifold structures between the source and target domains are different, allLapSVM, which applies LapSVM on the combination of the source and target domain data without adaption, results in poor classification accuracy. This implies that to use the labeled source domain data for classification in the target domain, a transformation of the source domain is necessary. The proposed semi-DA is able to learn a transformation for adaption as well as exploit the manifold structure from the unlabeled target domain data, thus can achieve the best classification performance.

We further conduct a third experiment by producing a set of 2-D two-moon data sets with varying distances. As shown in Fig. 3, all the comparison methods are insensitive to the change of distances. Not surprisingly, the semi-DA method produces the best results by exploiting unlabeled data, while DA performs on par or slightly better than tarLapSVM and tarSVM, which are then better performed than the rest methods.

For the last experiment, we consider a scenario where the two-moon points of the source domain are draw from a 3-D space, while the two-moon of the target domain resides in certain 2-D plane. This can be regarded as a version of the nonlinear extension case. As shown in Fig. 4, our method can still perform reasonably well under this scenario.

B. Cross-Domain Object Recognition

Our method is also applied to cross-domain object recognition using the benchmark data set in [14] and [15], which contains images from 31 object categories and three different image domains. The first domain contains product images downloaded from Amazon (denoted as amazon in the sequel), which are in a canonical pose with a white background. The second domain contains images that are taken with a digital SLR camera in an office (denoted as dslr). These images are high-resolution with varying poses and backgrounds. The third domain contains images, which are taken with a webcam using a flash, which are of low-resolution with varying poses

TABLE I
COMPARISON EXPERIMENTS ON THE TWO-MOON DATA SET IN TERMS OF ACCURACY (%)

Task	*srcSVM*	*tarSVM*	*allSVM*	*tarLapSVM*	*allLapSVM*	*DA*	*semi-DA*
src. 2-moon v.s. tar. 2-moon	51.9	63.5	51.9	69.5	51.7	67.0	**78.5**

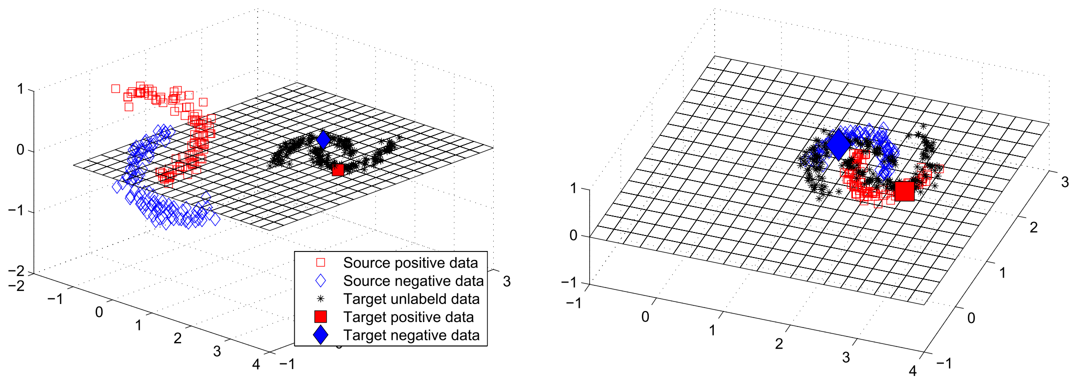


Fig. 4. Example of 3-D two-moon data set. (a) Source and target two-moon data sets. (b) Source and target two-moon data sets after applying the learned linear transformation.

and backgrounds (denoted as webcam). Six cross-domain object recognition tasks are considered: amazon versus dslr, amazon versus webcam, webcam versus dslr, webcam versus amazon, dslr versus amazon, dslr versus webcam, where the word before versus corresponds with the source domain and the word after versus corresponds with the target domain. As introduced in [14], all images are resized to the same width and converted to gray-scale, and are represented by SURF and SIFT features.

As shown in Table II, we compare our method (the semi-supervised log loss variant) with two state-of-the-art methods on this data set, namely Symm [14] and Asymm [15]. Both Symm and Asymm are based on metric learning with cross-domain constraints. Symm aims to learn a symmetric transformation to project the source and target domain data into a domain-invariant space, while Asymm aims to learn an asymmetric transformation to map the target domain data to the source domain. Both of them can be referred to as supervised domain adaptation methods. We use a radial basis function or RBF kernel for Symm and Asymm with width $\sigma = 1$ and use 1NN as the base classifier for Symm and Asymm because of its good performance reported in [14] and [15]. Overall, our method performs significantly better than the Symm method and on par with the Asymm method. Note both Symm and Asymm are dedicated methods [14], [15] for addressing the object recognition problem using this particular data set, while our proposed method is more general-purpose.

C. Cross-Domain WiFi Localization

We apply to the benchmark of cross-domain indoor WiFi localization [11]. The WiFi localization data set contains labeled WiFi data collected in two time period T_1 and T_2, which are considered as two different domains. Output labels are the corresponding locations where the WiFi data are received. Therefore, the label values are 2-D (in a floor) and continuous. In this data set, we only construct one cross-domain WiFi localization task T_1 versus T_2, because the data size of the domain T_1 is much smaller than that of the domain T_2. As the WiFi data are rather noisy, we first apply kernel principal component analysis or PCA using Laplace kernel with width $\sigma = 1$ on the WiFi data for denoising as proposed in [11]. The reduced dimensionality of the WiFi data is 20. The examples from target domain are further split into five disjoint folds. In each of the experiments, we alternately adopt one fold for training and the rest for testing. We also compare with our supervised domain adaptation variant, DA. The SSTCA method has been shown to be effective for cross-domain WiFi localization [11]. Following [11], we use a Laplace kernel with width $\sigma = 10$, set the reduced dimensionality of SSTCA to 20, and adopt a least square regression loss (i.e., Frobenius Norm).

TABLE II
COMPARISON OF OBJECT RECOGNITION ACCURACY (%)

Source	Target	*Symm*	*Asymm*	*semi-DA*
amazon	dslr	41.5	**58.2**	55.5
amazon	webcam	29.3	49.2	**51.4**
webcam	dslr	63.0	66.9	**68.3**
webcam	amazon	23.5	**30.9**	29.9
dslr	amazon	20.3	**30.9**	30.1
dslr	webcam	**62.6**	60.5	60.6

TABLE III
WIFI LOCALIZATION DATA SET WITH AVERAGE ERROR DISTANCE (UNIT: m)

Task Methods	*noTransf*	*SSTCA*	*DA*	*semi-DA*
T_1 vs T_2	2.25	1.89	1.81	**1.76**

TABLE IV
CLASSIFICATION ACCURACY (%) OF MULTISOURCE DOMAIN ADAPTATION ON SENTIMENT DATA SET

	SVM-C	*LWE*	*KE*	*KMM*	*SSTCA*	*DAM*	*2SW-MDA*	*semi-DA*
book	37.8	40.1	49.4	58.9	55.0	54.1	59.5	**63.5**
dvd	36.0	49.4	48.8	50.0	50.0	50.6	51.1	**56.1**
electronics	36.0	42.7	48.4	**65.6**	54.2	52.6	59.4	62.8
kitchen	35.6	40.1	49.4	64.0	64.1	58.6	**70.6**	68.6

As shown in Table III, our method DA already demonstrates superior performance on this data set. Moreover, it shows that further improvement has been achieved by semi-DA, which does not come as a surprise, as the unlabeled target instances are also used here.

D. Multisource Domain Adaptation for Sentiment Classification

So far, we have displayed the capacity of our algorithm to work with single-source domain adaptation problems. Here, we tackle the problem of multisource domain adaptation for sentiment classification, and compare with a recent method (2SW-MDA) [27] that requires a very little label information from the target domain. For this, we use the benchmark of [2] containing a collection of product reviews from Amazon. The reviews cover four product domains: books, dvds, electronics, and kitchen appliances. Each review has been annotated as negative or positive review sentiment polarity according to users' rating scores. In each domain, there are 1000 positive and 1000 negative reviews. Following the protocol of [27], the data set is preprocessed to reduce its feature dimension to 200, and only 1 labeled instance per class is used for target domain. Note that, the intention of this experiment is to make a fair comparison with existing state-of-the-art instead of solving the sentiment classification problem. On this problem, we compare our semi-DA method with (2SW-MDA) [27] as well as the suite of comparison methods reported in [27], including a SVM baseline (SVM-C), Locally weighted ensemble [28], kernel ensemble [29], kernel mean matching [30], SSTCA [11], and DAM [16]. The results of these comparison methods and the 2SW-MDA method are directly extracted from Table I of [27]. There are four tasks as presented in the rows of Table IV, where each task considers a classification scenario by adapting data from the rest three domains to the current domain. Overall, our method exhibits superior performance on two out of four tasks, and is shown to perform on par with [27] on the kitchen task. We attribute this performance gain to the ability of our approach to jointly learn the transformation function on a manifold and the prediction model.

V. OUTLOOK AND DISCUSSION

In this paper, we formulate the semi-supervised domain adaptation problem as learning with a transformation function and a prediction model under manifold constraints. We propose an iterative algorithm that jointly solves the manifold optimization problem by block coordinate descent with adaptive step-size. The algorithm can be easily adapted to work with different loss functions and prediction problems. Empirical evidence also supports the competitiveness of the proposed algorithm. For future work, we plan to develop a Newton-type update algorithm and to work with a broader range of practical domain adaptation problems.

APPENDIX

Our framework can work with a variety of loss functions; here, we discuss the square and log losses.

A. Square Loss

In this section, let X denote the target instance matrix of size $n \times d$, $\hat{X}$ the source instance matrix of size $\hat{n} \times d$, Φ the $d \times d$ linear transformation matrix, W a $d \times k$ weight matrix. Let Y and $\hat{Y}$ be the corresponding label matrices of size $n \times k$ and size $\hat{n} \times k$, respectively. Now, consider a square loss problem

$$\min_{\Phi,W} \frac{1}{2}\|W\|_F^2 + \frac{\eta_1}{2}\|XW - Y\|_F^2 + \frac{\eta_2}{2}\|\hat{X}\Phi W - \hat{Y}\|_F^2$$

subjected to constraints (3).

The reverse prediction framework of [31] provides good insights especially for cases where there exist unlabeled target instances. To start with, consider the fully labeled case and by (5) of [31], we have the following reverse prediction problem:

$$\min_{\Phi,U} \frac{1}{2}\|U\|_F^2 + \frac{\eta_1}{2}\|X - YU\|_F^2 + \frac{\eta_2}{2}\|\hat{X}\Phi - \hat{Y}U\|_F^2$$

subjected to constraints (3), where U is a matrix of $k \times d$.

Now, consider the unlabeled target examples, which can be denoted by X_U, a $(n-m) \times d$ matrix. Let the corresponding label be Z, a matrix of size $(n-m) \times k$. The forward prediction problem is

$$\min_{\Phi,W,Z} \frac{1}{2}\|W\|_F^2 + \frac{\eta_1}{2}\|XW - Y\|_F^2 + \frac{\eta_2}{2}\|\hat{X}\Phi W - \hat{Y}\|_F^2 + \eta_U\|X_U W - Z\|_F^2.$$

Note the third term is not helpful as Z can always be set to arbitrarily close to any W. Instead, we are interested in its reverse problem, which becomes that of

$$\min_{\Phi,U,Z} \frac{1}{2}\|U\|_F^2 + \frac{\eta_1}{2}\|X - YU\|_F^2 + \frac{\eta_2}{2}\|\hat{X}\Phi - \hat{Y}U\|_F^2 + \frac{\eta_U}{2}\|X_U - ZU\|_F^2$$

subjected to constraints (3).

1) Reverse Prediction: Consider the following reverse prediction problem:

$$\min_{\Phi,U} \ \frac{1}{2}\|U\|_F^2 + \frac{\eta_1}{2}\|X - YU\|_F^2 + \frac{\eta_2}{2}\|\hat{X}\Phi - \hat{Y}U\|_F^2 \quad (16)$$

subjected to (3), where U is a matrix of $k \times d$.

Consider the related loss function defined in (16), $l(\Phi) = 1/2\|\hat{X}\Phi - \hat{Y}U\|_F^2$, we have

$$\nabla_\Phi l(\Phi) = \hat{X}^\top(\hat{X}\Phi - \hat{Y}U). \quad (17)$$

By (1) and (17)

$$\tilde{\nabla}_\Phi l = \hat{X}^\top(\hat{X}\Phi - \hat{Y}U) - \Phi(\Phi^\top\hat{X}^\top - U^\top\hat{Y}^\top)\hat{X}\Phi. \quad (18)$$

Following (8) and (9), we have

$$P_t = \exp\left\{-\eta_2 \operatorname{skew}\left(\hat{X}^\top(\hat{X}\Phi - \hat{Y}U)\Phi^\top\right)\right\}. \quad (19)$$

Similarly, let $l(U) = 1/2\|X - YU\|_F^2 + \eta_2/2\eta_1\|\hat{X}\Phi - \hat{Y}U\|_F^2$. It is easy to check that

$$\nabla_U l(U) = \left(Y^\top Y + \frac{\eta_2}{\eta_1}\hat{Y}^\top\hat{Y}\right)U - \left(Y^\top X + \frac{\eta_2}{\eta_1}\hat{Y}^\top\hat{X}\Phi\right). \quad (20)$$

Similar to forward prediction, a possible initial guess of U can be $U = \left(Y^\top Y\right)^{-1}Y^\top X$.

2) Semi-supervised Learning: Now, consider the unlabeled target examples, which can be denoted by X_U, a $(n-m) \times d$ matrix. The optimization problem becomes that of

$$\min_{\Phi,U,Z} \ \frac{1}{2}\|U\|_F^2 + \frac{\eta_1}{2}\|X - YU\|_F^2 + \frac{\eta_2}{2}\|\hat{X}\Phi - \hat{Y}U\|_F^2 + \frac{\eta_3}{2}\|X_U - ZU\|_F^2 \quad (21)$$

subjected to (3). Note here X is a $m \times d$ matrix, Y a $m \times k$ matrix, and Z a $(n-m) \times k$ matrix. We have

$$\nabla_U l(U) = \left(Y^\top Y + \frac{\eta_2}{\eta_1}\hat{Y}^\top\hat{Y} + \frac{\eta_3}{\eta_1}Z^\top Z\right)U - \left(Y^\top X + \frac{\eta_2}{\eta_1}\hat{Y}^\top\hat{X}\Phi + \frac{\eta_3}{\eta_1}Z^\top X_U\right). \quad (22)$$

In addition, by adopting a similar strategy, we have $l(Z) = 1/2\|X_U - ZU\|_F^2$, so Z can also be computed as

$$\nabla_Z l(Z) = (ZU - X_U)U^\top \quad (23)$$

with an initial value $Z = X_U U^\top(UU^\top)^{-1}$.

B. Log Loss

Log loss is the loss function used in logistic regression problems.

1) Binary Classification:

$$\min_{w,\Phi} \ \eta_1\sum_{j=1}^{n}\log\left(1 + \exp\left(-y_i x_i^\top w\right)\right) + \eta_2\sum_{i=1}^{\hat{n}}\log\left(1 + \exp\left(-\hat{y}_i\hat{x}_i^\top\Phi w\right)\right) \quad (24)$$

subjected to (3). Here, $l_2(\Phi) := \sum_{i=1}^{\hat{n}}\log(1 + \exp(-\hat{y}_i\hat{x}_i^\top\Phi w))$, and gradient is

$$\nabla_\Phi l = -\sum_{i=1}^{\hat{n}}\frac{\hat{y}_i\hat{x}_i w^\top}{1 + \exp\left(\hat{y}_i\hat{x}_i^\top\Phi w\right)}. \quad (25)$$

Its manifold gradient is subsequently computed by (1).

On the other hand, we also have

$$l_1(w) := \eta_1\sum_{j=1}^{n}\log\left(1 + \exp\left(-y_j x_j^\top w\right)\right) + \eta_2\sum_{i=1}^{\hat{n}}\log\left(1 + \exp\left(-\hat{y}_i\hat{x}_i^\top\Phi w\right)\right) \quad (26)$$

and its gradient

$$\nabla_w l_1 = -\eta_1\sum_{j=1}^{n}\frac{y_j x_j}{1 + \exp\left(y_j x_j^\top w\right)} - \eta_2\sum_{i=1}^{\hat{n}}\frac{\hat{y}_i\Phi^\top\hat{x}_i}{1 + \exp\left(\hat{y}_i\hat{x}_i^\top\Phi w\right)}. \quad (27)$$

2) Multiclass Classification: For an (instance, label) pair (x, y), denote its feature function as $\gamma(x, y)$. The multiclass loss amounts to

$$\log\left(1 + \sum_{y' \neq y}\exp^{\gamma(x,y')^\top w - \gamma(x,y)^\top w}\right). \quad (28)$$

It can be shown that the gradient with respect to w is $\sum_{y'} p_x(y')\gamma(x, y') - \gamma(x, y)$, with $p_x(y') := \exp\gamma(x, y')^\top w / \sum_{y''}\gamma(x, y'')^\top w$, and the gradient with respect to Φ can be derived similarly.

In our setting, consider the objective function

$$\min_{w,\Phi} \ \eta_1\sum_{j=1}^{n}\log\left(1 + \sum_{y_j' \neq y_j}\exp^{\gamma(x_j,y_j')^\top w - \gamma(x_j,y_j)^\top w}\right) + \eta_2\sum_{i=1}^{\hat{n}}\log\left(1 + \sum_{\hat{y}_i' \neq \hat{y}_i}\exp^{\gamma(\hat{x}_i,\hat{y}_i')^\top\Phi w - \gamma(\hat{x}_i,\hat{y}_i)^\top\Phi w}\right) \quad (29)$$

subjected to $\Phi \in SO(d)$ (i.e., (3)). Here, $l_2(\Phi) := \sum_{i=1}^{\hat{n}}\log\left(1 + \sum_{\hat{y}_i' \neq \hat{y}_i}\exp^{\gamma(\hat{x}_i,\hat{y}_i')^\top\Phi w - \gamma(\hat{x}_i,\hat{y}_i)^\top\Phi w}\right)$, and gradient is

$$\nabla_\Phi l = \sum_{i=1}^{\hat{n}}\left(\sum_{\hat{y}_i'} p_{\hat{x}_i}(\hat{y}_i')\gamma(\hat{x}_i, \hat{y}_i')w^\top - \gamma(\hat{x}_i, \hat{y}_i)w^\top\right) \quad (30)$$

with $pp_{\hat{x}_i}(\hat{y}_i') := \exp\gamma(\hat{x}_i, \hat{y}_i')^\top\Phi w / \sum_{\hat{y}_i''}\gamma(\hat{x}_i, \hat{y}_i'')^\top\Phi w$. Its manifold gradient is subsequently computed by (1).

On the other hand, we also have

$$l_1(w) := \eta_1\sum_{j=1}^{n}\log\left(1 + \sum_{y_j' \neq y_j}\exp^{\gamma(x_j,y_j')^\top w - \gamma(x_j,y_j)^\top w}\right) + \eta_2\sum_{i=1}^{\hat{n}}\log\left(1 + \sum_{\hat{y}_i' \neq \hat{y}_i}\exp^{\gamma(\hat{x}_i,\hat{y}_i')^\top\Phi w - \gamma(\hat{x}_i,\hat{y}_i)^\top\Phi w}\right) \quad (31)$$

and its gradient being

$$\nabla_w l_1 = \eta_1 \sum_{j=1}^{n} \left(\sum_{y'_j} p_{x_j}(y'_j)\gamma(x_j, y'_j) - \gamma(x_j, y_j) \right) - \eta_2 \sum_{i=1}^{\hat{n}} \left(\sum_{\hat{y}'_i} p_{\hat{x}_i}(\hat{y}'_i)\gamma(\hat{x}_i, \hat{y}'_i) - \gamma(\hat{x}_i, \hat{y}_i) \right) \quad (32)$$

with $p_{x_j}(y'_j) := \exp \gamma(x_j, y'_j)^\top \Phi w / \sum_{y''_j} \gamma(x_j, y''_j)^\top \Phi w$.

Acknowledgment

The authors would like to thank B. Y. S. Khanloo for helpful discussions.

References

[1] H. Daumé, "Frustratingly easy domain adaptation," in *Proc. ACL*, 2007, pp. 256–263.

[2] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *Proc. ACL*, 2007, pp. 432–439.

[3] S. Pan, J. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *Proc. AAAI 23rd Nat. Conf. Artif. Intell.*, 2008, pp. 677–682.

[4] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. NIPS*, 2007, pp. 137–144.

[5] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, nos. 1–2, pp. 151–175, 2010.

[6] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[7] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proc. EMNLP*, 2006, pp. 120–128.

[8] R. 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, Dec. 2005.

[9] H. Daumé, A. Kumar, and A. Saha, "Co-regularization based semi-supervised domain adaptation," in *Proc. NIPS*, 2010, pp. 478–486.

[10] J. Donahue, J. Hoffman, E. Rodner, K. Saenko, and T. Darrell, "Semi-supervised domain adaptation with instance constraints," in *Proc. IEEE CVPR*, Jun. 2013, pp. 668–675.

[11] S. Pan, I. Tsang, J. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[12] G. Skolidis and G. Sanguinetti, "Semisupervised multitask learning with Gaussian processes," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 12, pp. 2101–2112, Dec. 2013.

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

[14] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. 11th ECCV*, 2010, pp. 213–226.

[15] B. Kulis, K. Saenko, and T. Darrell, "What you saw is not what you get: Domain adaptation using asymmetric kernel transforms," in *Proc. IEEE Conf. CVPR*, Jun. 2011, pp. 1785–1792.

[16] L. Duan, I. Tsang, D. Xu, and T. Chua, "Domain adaptation from multiple sources via auxiliary classifiers," in *Proc. 26th Annu. ICML*, 2009, pp. 289–296.

[17] A. Edelman, T. Arias, and S. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM J. Matrix Anal. Appl*, vol. 20, no. 2, pp. 303–353, 1998.

[18] J. Lee, *Introduction to Smooth Manifolds*. New York, NY, USA: Springer-Verlag, 2003.

[19] P. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ, USA: Princeton Univ. Press, 2008.

[20] 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.

[21] D. Bertsekas, *Convex Analysis and Optimization*. Belmont, MA, USA: Athena Scientific, 2003.

[22] K. Tsuda, G. Ratsch, and M. Warmuth, "Matrix exponentiated gradient updates for on-line learning and Bregman projections," *J. Mach. Learn. Res.*, vol. 6, pp. 995–1018, Dec. 2005.

[23] C. Kelley, *iterative methods for optimization*. Philadelphia, PA, USA: SIAM, 1999.

[24] Y. Yang, "Globally convergent optimization algorithms on Riemannian manifolds: Uniform framework for unconstrained and constrained optimization," *J. Optim. Theory Appl.*, vol. 132, no. 2, pp. 245–265, 2007.

[25] N. Higham, "The scaling and squaring method for the matrix exponential revisited," *SIAM J. Matrix Anal. Appl.*, vol. 26, no. 4, pp. 1179–1193, 2005.

[26] C. Moler and C. Loan, "Nineteen dubious ways to compute the exponential of a matrix," *SIAM Rev.*, vol. 20, no. 4, pp. 801–836, 1978.

[27] Q. Sun, R. Chattopadhyay, S. Panchanathan, and J. Ye, "A two-stage weighting framework for multi-source domain adaptation," in *Proc. NIPS*, 2011, pp. 505–513.

[28] J. Gao, W. Fan, J. Jiang, and J. Han, "Knowledge transfer via multiple model local structure mapping," in *Proc. 14th KDD*, 2008, pp. 283–291.

[29] E. Zhong *et al.*, "Cross domain distribution adaptation via kernel mapping," in *Proc. 15th KDD*, 2009, pp. 1027–1036.

[30] J. Huang, A. Gretton, B. Scholkopf, A. Smola, and K. Borgwardt, "Correcting sample selection bias by unlabeled data," in *Proc. NIPS*, 2007, pp. 601–608.

[31] L. Xu, M. White, and D. Schuurmans, "Optimal reverse prediction: A unified perspective on supervised, unsupervised and semi-supervised learning," in *Proc. ICML*, 2009, pp. 143–151.

Li Cheng (M'00–S'01–M'04–SM'11) received the Ph.D. degree in computer science from the University of Alberta, Edmonton, AB, Canada.

He is a Research Scientist with the Bioinformatics Institute (BII). Prior to joining BII in 2010, he was with the Statistical Machine Learning Group, National ICT Australia Ltd., Sydney, Australia, the Toyota Technological Institute at Chicago, Chicago, IL, USA, and the University of Alberta. His current research interests include machine learning and computer vision.

Sinno Jialin Pan received the Ph.D. degree in computer science from the Hong Kong University of Science and Technology, Hong Kong, in 2010.

He is currently a Scientist and Laboratory Head of Text Analytics with the Data Analytics Department, Institute for Infocomm Research, Singapore. He holds a position of Adjunct Assistant Professor with the Department of Computer Science, School of Computing, National University of Singapore, Singapore. He has published more than 30 research papers in leading journals and top conference proceedings in multidisciplinary areas, including the IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Neural Networks, IEEE Transactions on Knowledge and Data Engineering, *ACM Transactions on Information Systems*, IEEE Transactions on Computational Biology and Bioinformatics/*ACM Transactions on Computational Biology and Bioinformatics*, *Journal of Artificial Intelligence*, Association for the Advancement of Artificial Intelligence, SIGKDD, International Joint Conference on Artificial Intelligence, Association for Computational Linguistics, World Wide Web, International Conference on Software Engineering, and Ubiquitous Cmputing. His current research interests include transfer learning and its applications to wireless-sensor-based data mining, text mining, sentiment analysis, software engineering, and bioinformatics.

Dr. Pan served as the Workshop Co-Chairs of the NIPS Workshop on Transfer Learning for Structured Data in 2009 and the ICDM Workshop on Transfer Mining in 2009. He also served as a Co-Guest Editor of the *ACM Transactions on Intelligent Systems and Technology* on the Domain Adaptation in Nature Language Processing special issue.