Pattern Recognition 75 (2018) 235–249

Contents lists available at ScienceDirect

Pattern Recognition

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

Semi-supervised transfer subspace for domain adaptation

Luís A.M. Pereira*, Ricardo da Silva Torres

Institute of Computing, University of Campinas – UNICAMP, São Paulo, Brazil

ARTICLE INFO

Article history:
Received 16 November 2016
Revised 19 February 2017
Accepted 12 April 2017
Available online 18 April 2017

Keywords:
Cross-domain knowledge transfer
Cross-dataset classification
Dataset bias
Metric learning
Semi-supervised learning

ABSTRACT

Domain shift is defined as the mismatch between the marginal probability distributions of a source (training set) and a target domain (test set). A successful research line has been focusing on deriving new source and target feature representations to reduce the domain shift problem. This task can be modeled as a semi-supervised domain adaptation. However, without exploiting at the same time the knowledge available on the labeled source, labeled target, and unlabeled target data, semi-supervised methods are prone to fail. Here, we present a simple and effective Semi-Supervised Transfer Subspace (SSTS) method for domain adaptation. SSTS establishes pairwise constraints between the source and labeled target data, besides it exploits the global structure of the unlabeled data to build a domain invariant subspace. After reducing the domain shift by projecting both source and target domain onto this subspace, any classifier can be trained on the source and tested on target. Results on 49 cross-domain problems confirm that SSTS is a powerful mechanism to reduce domain shift. Furthermore, SSTS yields better classification accuracy than state-of-the-art domain adaptation methods.

1. Introduction

Domain shift is defined as the mismatch between the marginal probability distributions of a source (training set) and a target domain (test set). This is a prevalent problem in machine learning mainly in real-world applications. In computer vision, for instance, domain shift occurs essentially because visual data are often captured by different devices and under varied imaging conditions such as scene, pose, and illumination [30]. Across visual datasets the domain shift – also known as dataset bias – can be severe [35] (Fig. 1). Speech and language processing are also subjected to domain shift [3,9]. Under such scenarios, however, conventional classifiers often fail to achieve desirable performances at test time [30] because they assume a stationary environment, *i.e.*, source and target domain are supposedly drawn from the same probability distribution. The limitation of this assumption has motivated the development of domain adaptation methods to reduce the domain shift and increase the classifier performance (refer to [16,29] for a complete literature review on domain adaptation).

Such methods are proposed either in semi-supervised or in unsupervised settings. The semi-supervised methods use a fully labeled source data and a partially labeled target data to guide the domain adaptation (*e.g.*, [30,36,38,40]). In contrast, unsupervised methods use a fully labeled source along with a fully unlabeled target data (*e.g.*, [5,12,14,22,23]). Usually, both fully labeled source and fully unlabeled target data are in plenty. Labeled target data, in turn, are more scarce. Generally, semi-supervised methods perform better than unsupervised methods even in the presence of few amount of labeled target data [14,15] – besides, better results can be achieved if the unlabeled target data is properly exploited [36,38,40].

Several semi-supervised domain adaptation methods were proposed as max-margin classifier extensions (*e.g.*, [4,11,18]), whose goal is to learn the model parameters on the source domain and then transfer them to the target domain. The assumption is that the target model is a perturbed version of the source model. Without enough representative labeled target instances, these methods are prone to perform poorly due to the importance given to class labels. Hence, several studies advocate for feature-based methods, whose goal is to reduce the domain shift by approximating the source and target feature distributions [29]; therefore, any classifier can be trained on the source domain and tested on the target domain.

A prominent feature-based method (which inspired several works including ours) was proposed by Saenko et al. [30]. The method uses information theoretic metric learning to learn a regularized transformation, *i.e.*, an explicit domain-invariant metric. This metric is then applied to map source features to target features. A key concept of this algorithm is to establish inter-domain pairwise constraints between the source and labeled target in-

* Corresponding author.
E-mail addresses: luis.pereira@ic.unicamp.br (L.A.M. Pereira), rtorres@ic.unicamp.br (R. da S. Torres).

http://dx.doi.org/10.1016/j.patcog.2017.04.011
0031-3203/

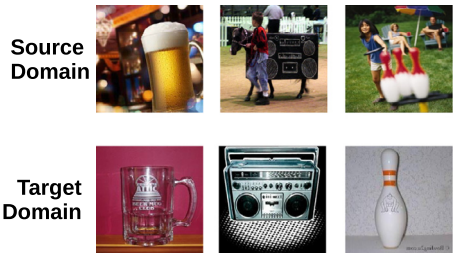

Fig. 1. Source (training set) and the target (test set) domain may contain images belonging to same class though captured under different scenes, pose, and/or illumination. This imaging conditions cause mismatch (dataset bias) between the source and target distributions.

stances to preserve the target discriminative structure. Unlike other feature-based algorithms, this method does not take any advantage of unlabeled target data (generally in plenty). This may justify its poor performance when compared to subspace-based algorithms (*e.g.*, [14,15]), which somehow exploit the information available on the unlabeled target data.

Semi-supervised subspace-based methods are feature-based algorithms designed under the assumption of a common domain-invariant subspace (*i.e.*, an implicit domain-invariant metric) in which the domain shift between source and target distributions is reduced. The state-of-the-art subspace-based methods exploit the information available on both source and target domain. The Laplacian Embedding framework is the common choice to exploit the underlying local structure of the unlabeled target data (*e.g.*, [6,26,36,39,40]). Despite its theoretical appeal, such approach is graph-dependent, *i.e.*, the better the graph is induced, the better it captures the data local structure; nevertheless, the opposite is also true [1,37]. Determining the best graph type (*e.g.*, k-nn or ϵ-ball) is a complex task [37,41] and, depending on the choice, the complexity of the domain adaptation can increase and its performance can also be negatively affected.

Here, we present a simple and effective semi-supervised method for domain adaptation. Our method – referred to as Semi-Supervised Transfer Subspace (SSTS) – exploits properties of the source and labeled/unlabeled target data to yield a domain-invariant subspace. Essentially, SSTS uses two processes to accomplish this task. First, SSTS takes advantage of the data global structure, (*i.e.*, the data variance), including the target unlabeled data. This allows enhancing the domain adaptation [14,15] without the need of handling the inconveniences of the local structure preserving approach. Second, SSTS establishes interdomain pairwise constraints between the source and labeled target instances to preserve discriminative properties (*i.e.*, the classes separability) in the domain-invariant subspaces. Mathematically, the combination of these two processes leads to a (non)linear domain-invariant metric (as we show in Section 5). After projecting both source and target domain onto the domain-invariant subspaces to reduce the domain shift, any classifier can be trained on the source and tested on target. A schematic illustration of SSTS is displayed in Fig. 2.

We carried out extensive experiments on 49 real-world visual cross-domain problems. Besides the standard manually-labeled cross-domain problems, we also evaluated SSTS in the weakly-labeled scenario, which has Internet photos retrieved by keyword-based image search engines. Results on these cross-domain problems confirm that the SSTS is a powerful mechanism to reduce domain shift. Furthermore, SSTS yields better classification accuracy than state-of-the-art domain adaptation methods.

2. Notations

We consider that our data come from two domains:

(i) a fully labeled source domain $X^{\mathcal{S}} = [\boldsymbol{x}_1^{\mathcal{S}}, \boldsymbol{x}_2^{\mathcal{S}}, \ldots, \boldsymbol{x}_{n_{\mathcal{S}}}^{\mathcal{S}}] \in \mathbb{R}^{\mathcal{D}\times n_{\mathcal{S}}}$; and

(ii) a partially labeled target domain with its labeled part denoted as $X^{\mathcal{T}} = [\boldsymbol{x}_1^{\mathcal{T}}, \boldsymbol{x}_2^{\mathcal{T}}, \ldots, \boldsymbol{x}_{n_{\mathcal{T}}}^{\mathcal{T}}] \in \mathbb{R}^{\mathcal{D}\times n_{\mathcal{T}}}$, and the unlabeled part denoted as $X_{\mathcal{U}}^{\mathcal{T}} = [\boldsymbol{x}_1^{\mathcal{U}}, \boldsymbol{x}_2^{\mathcal{U}}, \ldots, \boldsymbol{x}_{n_{\mathcal{U}}}^{\mathcal{U}}] \in \mathbb{R}^{\mathcal{D}\times n_{\mathcal{U}}}$, with $n_{\mathcal{T}} \ll n_{\mathcal{S}}$ and $n_{\mathcal{T}} \ll n_{\mathcal{U}}$;

such that $P(X^{\mathcal{S}}) \neq P(X^{\mathcal{T}} \cup X_{\mathcal{U}}^{\mathcal{T}})$, while $P(Y^{\mathcal{S}}|X^{\mathcal{S}}) \approx P(Y^{\mathcal{T}}|X^{\mathcal{T}} \cup X_{\mathcal{U}}^{\mathcal{T}})$ with $Y^{\mathcal{S}}$ and $Y^{\mathcal{T}}$ standing for the source and the target set of labels, respectively. Additionally, let $\tilde{\boldsymbol{X}} = [X^{\mathcal{S}}|X^{\mathcal{T}}|X_{\mathcal{U}}^{\mathcal{T}}] = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N] \in \mathbb{R}^{\mathcal{D}\times N}$ be the concatenated matrix of instances from both domains with $N = n_{\mathcal{S}} + n_{\mathcal{T}} + n_{\mathcal{U}}$. Note that an instance $\boldsymbol{x} \in \tilde{\boldsymbol{X}}$ is a column vector denoted without any superscript.

Let $\mathcal{M}, \mathcal{C} \subset X^{\mathcal{S}} \times X^{\mathcal{T}}$ be two sets of pairwise constraints such that

- $\mathcal{M} = \{(\boldsymbol{x}_i^{\mathcal{S}}, \boldsymbol{x}_j^{\mathcal{T}}) \,|\, \boldsymbol{x}_i^{\mathcal{S}}$ and $\boldsymbol{x}_j^{\mathcal{T}}$ are similar$\}$ is the set of inter-domain must-link constraints; and
- $\mathcal{C} = \{(\boldsymbol{x}_i^{\mathcal{S}}, \boldsymbol{x}_j^{\mathcal{T}}) \,|\, \boldsymbol{x}_i^{\mathcal{S}}$ and $\boldsymbol{x}_j^{\mathcal{T}}$ are dissimilar$\}$ is the set of inter-domain cannot-link constraints.

The words "similar" and "dissimilar" mean "same class," and "different class," respectively.

3. Description of the proposed method

Here, we describe in detail the SSTS linear version. In Section 4, we show how this linear version can be extended to handle non-linear feature deformations.

3.1. Encoding discriminative knowledge

To exploit the discriminative knowledge available in the source domain (in large amount) and in the target domain (in a small amount), inter-domain pairwise constraints were encoded into two different functionals. The first functional concerns to *maximize* the squared induced distance between instances belonging to the *different* class and from *different* domains. Mathematically, this functional is defined as

$$\mathcal{F}_{\mathcal{C}}(Q) = \frac{1}{2} \sum_{\forall (x_i^{\mathcal{S}}, x_j^{\mathcal{T}}) \in \mathcal{C}} W_{ij}^{\mathcal{C}} ||Q^{\top}\boldsymbol{x}_i^{\mathcal{S}} - Q^{\top}\boldsymbol{x}_j^{\mathcal{T}}||_2^2, \tag{1}$$

in which

$$W_{ij}^{\mathcal{C}} = \begin{cases} \dfrac{1}{|\mathcal{C}|} & \text{if } (\boldsymbol{x}_i^{\mathcal{S}}, \boldsymbol{x}_j^{\mathcal{T}}) \in \mathcal{C}, \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

is the matrix of weights for cannot-link constraints. $Q \in \mathbb{R}^{\mathcal{D}\times d}$ is the transformation matrix.

Simplifying the functional in Eq. (1) implies that

$$\mathcal{F}_{\mathcal{C}}(Q) = \mathrm{Tr}(Q^{\top}\tilde{\boldsymbol{X}}\mathcal{L}_{\mathcal{C}}\tilde{\boldsymbol{X}}^{\top}Q), \tag{3}$$

in which $\mathrm{Tr}(\cdot)$ stands for the trace operator.[1] The matrix $\mathcal{L}_{\mathcal{C}} \equiv D^{\mathcal{C}} - W^{\mathcal{C}}$ is the Laplacian matrix [7], such that $D^{\mathcal{C}}$ is a diagonal matrix whose elements in the principal diagonal are defined as $D_{ii}^{\mathcal{C}} = \sum_j W_{ij}^{\mathcal{C}}$ (i.e., the column sum of the matrix $W^{\mathcal{C}}$).

The second functional attempts to *minimize* the squared induced distance between instances belonging the *same class* and from *different* domains. Analytically, this functional is defined as

$$\mathcal{F}_{\mathcal{M}}(Q) = \frac{1}{2} \sum_{\forall (x_i^{\mathcal{S}}, x_j^{\mathcal{T}}) \in \mathcal{M}} W_{ij}^{\mathcal{M}} ||Q^{\top}\boldsymbol{x}_i^{\mathcal{S}} - Q^{\top}\boldsymbol{x}_j^{\mathcal{T}}||_2^2, \tag{4}$$

[1] Note that the superscript $^{\top}$ denotes the transpose operator, while the superscript $^{\mathcal{T}}$ indicates the target domain.

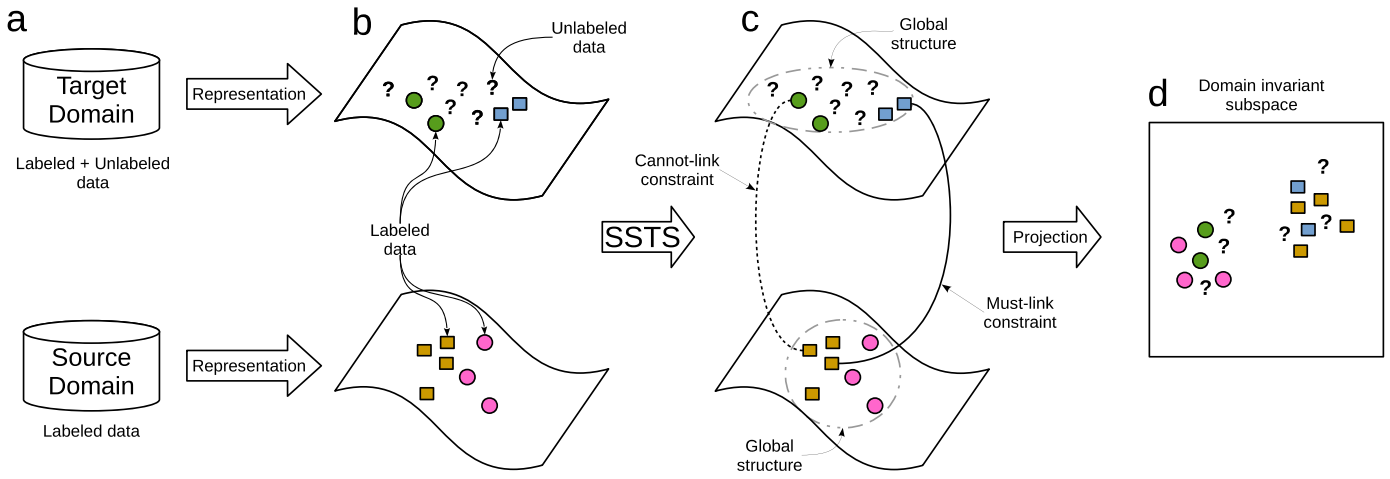

Fig. 2. The basic idea of SSTS. (a) The data come from a partially label target and a fully labeled source domain. (b) Both source and target domains must be represented into the same feature space: the target domain contains a few labeled instances (green circles and blue squares) and unlabeled instances in its majority (depicted as "**?**"); the source domain contains only labeled instances (pink circles and yellow squares). (c) SSTS establishes must-link constraints between instances belonging to the same class and from different domains (*e.g.*, the link between the yellow square and the blue square) and cannot-link constraints instances belonging to different classes and different domains (*e.g.*, the link between the yellow square and the green circle). SSTS also exploits the data global structure (variance). (d) After SSTS builds the domain-invariant subspace, both source and target domains are projected onto the such a subspace. It is expected that similar instances stay close whilst dissimilar instances stay far apart. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

in which

$$W_{ij}^{\mathcal{M}} = \begin{cases} \dfrac{1}{|\mathcal{M}|} & \text{if } (\boldsymbol{x}_i^S, \boldsymbol{x}_j^T) \in \mathcal{M}, \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

is the matrix of weights for must-link constraints.

Again, simplifying the functional above (Eq. (4)) implies that

$$\mathcal{F}_{\mathcal{M}}(Q) = \mathrm{Tr}(Q^\top \tilde{\boldsymbol{X}} \mathcal{L}_{\mathcal{M}} \tilde{\boldsymbol{X}}^\top Q). \tag{6}$$

$\mathcal{L}_{\mathcal{M}} \equiv D^{\mathcal{M}} - W^{\mathcal{M}}$ is also a Laplacian matrix, with $D^{\mathcal{M}}$ a diagonal matrix whose elements in the principal diagonal are defined as $D_{ii}^{\mathcal{M}} = \sum_j W_{ij}^{\mathcal{M}}$ (i.e., the column sum of the matrix $W^{\mathcal{M}}$).

The rationale behind these two functional ($\mathcal{F}_{\mathcal{C}}$ and $\mathcal{F}_{\mathcal{M}}$) is to rearrange source and target instances such that *dissimilar* (different classes) pair of instances stay away, whereas *similar* (same classes) pair of instances stay close (see Fig. 2). In Section 3.3, we show how these functional are combined to achieve this goal.

3.2. Encoding global structure knowledge

Depending on the domain-shift level, the labeled target data by itself (generally in a small amount) may be insufficient to guide the domain adaptation process, that may lead to poor classification performance of the target domain. Hence, solutions to harness effectively the unlabeled target data are important to enhance the semi-supervised domain adaptation performance [14,20]. To take advantage of these data, the following functional is employed:

$$\mathcal{F}(Q) = \frac{1}{2} \sum_{x_i \in \tilde{\boldsymbol{X}}} \sum_{x_j \in \tilde{\boldsymbol{X}}} W_{ij} ||Q^\top \boldsymbol{x}_i - Q^\top \boldsymbol{x}_j||_2^2. \tag{7}$$

Simplifying $\mathcal{F}$ implies that

$$\mathcal{F}(Q) = \mathrm{Tr}(Q^\top \tilde{\boldsymbol{X}} \mathcal{L} \tilde{\boldsymbol{X}}^\top Q), \tag{8}$$

in which $\mathcal{L} \equiv D - W$ is a Laplacian matrix for $W_{ij} = 1/N^2$ and $D_{ii} = \sum_j W_{ij} = I/N$ is a diagonal matrix, I is the identity matrix. Rewriting the Laplacian matrix, we have that $\mathcal{L} \equiv \frac{1}{N} I - \frac{1}{N^2} \mathbf{1}\mathbf{1}^\top$, in which $\mathbf{1}$ is a column vector of N ones. It is simple to show that if $W_{ij} = 1/N^2$, then the product $\tilde{\boldsymbol{X}} \mathcal{L} \tilde{\boldsymbol{X}}^\top$ is equal to the data covariance matrix, *i.e.*, $\tilde{\boldsymbol{X}} \mathcal{L} \tilde{\boldsymbol{X}}^\top = E[(\boldsymbol{x} - \bar{\boldsymbol{x}})(\boldsymbol{x} - \bar{\boldsymbol{x}})^\top]$, in which $\bar{\boldsymbol{x}} = \frac{1}{N} \sum_i \boldsymbol{x}_i$ is the sample mean [17].

Specifically, $\mathcal{L}$ is equivalent to the centering matrix, *i.e.*, it removes the sample mean from the instance vectors. Therefore, the functional $\mathcal{F}$ expresses the average squared distance between all $\boldsymbol{x}_i \in \tilde{\boldsymbol{X}}$ in the subspace, which is equivalent to the Principal Component Analysis (PCA) criterion. Thus, the rationale behind $\mathcal{F}$ is to exploit the global structure knowledge through the cross-domain data covariance.

3.3. Objective function

Now we have the three parts needed to compose the SSTS objective function: $\mathcal{F}$, $\mathcal{F}_{\mathcal{C}}$, and $\mathcal{F}_{\mathcal{M}}$. The goal, then, is to combine these three functionals to yield a domain adaptation approach, which takes advantage of the inter-domain pairwise constraints defined between the source and target labeled data and the global structure knowledge provided by the target unlabeled data. To accomplish this goal, the objective function was designed as the generalized Rayleigh quotient [8]

$$\arg\max_{Q} \frac{\mathcal{F}_{\mathcal{C}}(Q) + \mathcal{F}(Q)}{\mathcal{F}_{\mathcal{M}}(Q)}, \tag{9}$$

which can be represented by

$$\begin{aligned} \arg\max_{Q} \quad & \mathrm{Tr}(Q^\top \tilde{\boldsymbol{X}} (\mathcal{L}_{\mathcal{C}} + \gamma \mathcal{L}) \tilde{\boldsymbol{X}}^\top Q) \\ \text{subject to} \quad & Q^\top \tilde{\boldsymbol{X}} \mathcal{L}_{\mathcal{M}} \tilde{\boldsymbol{X}}^\top Q = I \text{ and} \\ & Q^\top Q = I, \end{aligned} \tag{10}$$

with γ a hyper-parameter to control the contribution of $\mathcal{L}$. The constraints disallow the solution to collapse to the trivial solution (i.e., where all distances are zero), avoid a rank one solution, assure that the different coordinates in the feature space are uncorrelated, and also ensure the positive definite properties.

The Lagrangian of the optimization problem in Eq. (10) is

$$\begin{aligned} \mathscr{L}(Q, \Lambda, \beta) &= Q^\top \tilde{\boldsymbol{X}} (\mathcal{L}_{\mathcal{C}} + \gamma \mathcal{L}) \tilde{\boldsymbol{X}}^\top Q \\ &\quad - \Lambda (Q^\top \tilde{\boldsymbol{X}} \mathcal{L}_{\mathcal{M}} \tilde{\boldsymbol{X}}^\top Q - I) - \beta (Q^\top Q - I) \\ \nabla \mathscr{L}(Q, \Lambda, \beta) &= \tilde{\boldsymbol{X}} (\mathcal{L}_{\mathcal{C}} + \gamma \mathcal{L}) \tilde{\boldsymbol{X}}^\top Q - \Lambda \tilde{\boldsymbol{X}} \mathcal{L}_{\mathcal{M}} \tilde{\boldsymbol{X}}^\top Q - \beta Q = 0. \end{aligned} \tag{11}$$

Therefore, the SSTS optimization problem amounts to solve

$$\tilde{\boldsymbol{X}} (\mathcal{L}_{\mathcal{C}} + \gamma \mathcal{L}) \tilde{\boldsymbol{X}}^\top Q = \Lambda (\tilde{\boldsymbol{X}} \mathcal{L}_{\mathcal{M}} \tilde{\boldsymbol{X}}^\top + \beta I) Q \tag{12}$$

or

$$\{\tilde{\boldsymbol{X}}\mathcal{L}_{\mathcal{M}}\tilde{\boldsymbol{X}}^{\top}+\beta I\}^{-1}\{\tilde{\boldsymbol{X}}(\mathcal{L}_{\mathcal{C}}+\gamma\mathcal{L})\tilde{\boldsymbol{X}}^{\top}\}Q=\Lambda Q \tag{13}$$

which is a typical eigenvalue problem [8] solved efficiently by eigendecomposition. The eigenvectors in Q, corresponding to the d largest eigenvalues in Λ, form an orthogonal basis of the domain-invariant subspace.

4. Extension to nonlinear problems

There are certain problems whose domains have strongly nonlinear manifolds. In such settings, however, SSTS may fail to capture the intrinsic data geometry because it assumes linear subspaces. To approach this shortcoming, here we present an SSTS nonlinear extension based on Reproducing Kernel Hilbert Space (RKHS). Basically, the idea is to find a mapping from the nonlinear input space to another space where linear methods can be safely applied. This concept is extensively used in the field of Kernel Machines [32] (*e.g.*, Support Vector Machines) and also to design nonlinear version of linear dimensionality reduction methods (*e.g.*, [24,31]).

Let

$$\begin{aligned}\phi &: \mathbb{R}^{\mathcal{D}} \rightarrow \mathcal{H}\\ \boldsymbol{x} &\mapsto \phi(\boldsymbol{x})\end{aligned} \tag{14}$$

be a nonlinear mapping from a Euclidean space $\mathbb{R}^{\mathcal{D}}$ to a Hilbert space $\mathcal{H}$ with $\boldsymbol{x}\in\tilde{\boldsymbol{X}}$, such that $\Phi=[\phi(\boldsymbol{x}_1),\phi(\boldsymbol{x}_2),\ldots,\phi(\boldsymbol{x}_N)]$ is the data in the RKHS.

Choosing ϕ as symmetric positive definite allows us to define the kernel function

$$\begin{aligned}k &: \mathcal{D}\times\mathcal{D}\rightarrow\mathbb{R}\\ \boldsymbol{x}_i,\boldsymbol{x}_j &\mapsto k(\boldsymbol{x}_i,\boldsymbol{x}_j),\end{aligned} \tag{15}$$

in which

$$k(\boldsymbol{x}_i,\boldsymbol{x}_j)=\langle\phi(\boldsymbol{x}_i),\phi(\boldsymbol{x}_j)\rangle_{\mathcal{H}}=\phi(\boldsymbol{x}_i)^{\top}\phi(\boldsymbol{x}_j). \tag{16}$$

With this function, the dot product in $\mathcal{H}$ can be computed implicitly, *i.e.*, without the need of mapping the vectors before computing dot products, what is popularly known as "kernel trick." Associated with k, we can define the Gram matrix $K\equiv\Phi^{\top}\Phi$, in which $K_{ij}=k(\boldsymbol{x}_i,\boldsymbol{x}_j)$.

By combining the exposition above with the Representer theorem, the optimization in Eq. (10) can be rewritten as follows:

$$\begin{aligned}&\Phi^{\top}\Phi(\mathcal{L}_{\mathcal{C}}-\gamma\mathcal{L})\Phi^{\top}\Phi V=\Lambda\Phi^{\top}\Phi(\mathcal{L}_{\mathcal{M}}+\beta I)\Phi^{\top}\Phi V\\ &K(\mathcal{L}_{\mathcal{C}}-\gamma\mathcal{L})KV=\Lambda K(\mathcal{L}_{\mathcal{M}}+\beta I)KV\end{aligned} \tag{17}$$

or

$$\{K(\mathcal{L}_{\mathcal{M}}+\beta I)K\}^{-1}\{K(\mathcal{L}_{\mathcal{C}}-\gamma\mathcal{L})K\}V=\Lambda V \tag{18}$$

Similarly to the linear version, the optimization above is an eigenvalue problem. The eigenvectors in V, corresponding to the d largest eigenvalues in Λ, form the domain-invariant subspace basis.

5. Learning a classifier

The advantage of encoding the domain invariance into the transformation matrices Q (linear version) or V (nonlinear version) is that a broad range of classifiers can be employed to classify the target domain. For instance, a distance-based model can be easily trained by considering the distance between any two instances $\boldsymbol{x}_i,\boldsymbol{x}_j\in\tilde{\boldsymbol{X}}$ given by

$$f(\boldsymbol{x}_i,\boldsymbol{x}_j|M)=\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^{\top}M(\boldsymbol{x}_i-\boldsymbol{x}_j)}, \tag{19}$$

in which $M=QQ^{\top}$ is a definite semi-positive matrix.

6. Experimental design

Here we present the datasets, the baselines and the statistical analysis used in the experiments.

6.1. Cross-domain problems

In what follows, we describe five sets of cross-domain image problems employed in our experiments, as well as the experimental methodology related to them. These sets are extensively used as domain adaptation benchmarks (*e.g.*, [14,30,36]) due to their visual properties, which induce different levels of domain shift across their domains. Their main quantitative aspects are displayed in Table 1.

Office+Caltech10 [14,30] is a cross-domain set composed of four existing object image dataset (or domains) namely, Amazon, Caltech10, DSLR, and Webcam. The images were captured by cameras of different resolution quality, downloaded from online merchants and from Google, which induces different levels of domain shift across these domains (Fig. 3). All combination pairs of them were evaluated as a cross-domain problem, using one domain as source and the other as the target. Note that the experiments were performed following the standard protocol used to evaluate the Office+Caltech10 set in a semi-supervised cross-domain scenario (*e.g.*, [14,30,36]). Additionally, to better understand how SSTS works on Office+Caltech10, two different image description procedures were used:

1. A normalized histogram of visual words was obtained from a codebook constructed from a subset of the Amazon dataset on points of interest detected by the Speeded Up Robust Features (SURF) method[2] (refer to [30] for further details);
2. Features were extracted as the sparse activations of the fully connected 7th layer of a convolutional neural network trained on the ImageNet dataset and then fine tuned on the four domains from Office+Caltech10[3] (refer to [10] for further details).

COIL20 (**C**olumbia **O**bject **I**mage **L**ibrary) is a dataset composed of gray level images of 20 objects. The objects were captured in 72 different poses by a fixed camera and a 360-degrees motorized turntable which varied the object pose in steps of 5° [25].[4] To induce a domain-shit in this dataset, two subsets (domains) – hereafter referred to as COIL-A and COIL-B – were sampled from COIL20. COIL-A was composed of images taken in the directions of $[0°, 85°]\cup[180°, 265°]$, which correspond to the first and third quadrants, whereas COIL-B was composed of images taken in the directions of $[90°, 175°]\cup[270°, 355°]$, which correspond to the second and fourth quadrants (Fig. 4). Then, two cross-domain problems were constructed using (i) COIL-A as source and COIL-B as target domain; and (ii) COIL-B as source and COIL-A as target domain. All image was rescaled to 32×32 and its gray-scale pixel values used as image representation. This methodology was also used in [21].

Digits is a cross-domain set composed of two existing image datasets of handwritten digits namely, MNIST[5] and USPS[6], which share 10 classes of digits. A domain shift across MNIST and USPS is considered due to the different gray level distribution of these

[2] Office+Caltech10 with DeCAF features is available at https://www.eecs.berkeley.edu/jhoffman/domainadapt/ (as of 12/09/2016).

[3] Office+Caltech10 with DeCAF features is available at https://github.com/dtuia/KEMA.git (as of 12/09/2016).

[4] COIL20 is available at http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php (as of 12/09/2016).

[5] MNIST is available at http://yann.lecun.com/exdb/mnist/ (as of 12/09/2016).

[6] USPS is available at http://www-i6.informatik.rwth-aachen.de/~keysers/usps.html (as of 12/09/2016).

Table 1
Quantitative aspects of the cross-domain sets and their domains.

Sets	Domains	#samples[a]	#features	#classes	#cross-domain problems
Bing-Caltech256	Bing Caltech256	2560 7680	2625	256	1
COIL20	COIL-A COIL-B	720 720	1024	20	2
Digits	MNIST USPS	2000 1800	256	10	2
Office+Caltech10[b]	Amazon Caltech10 DSLR Webcam	958 1123 157 295	800 (SURF), 4096 (DeCAF)	10	12 (SURF), 12 (DeCAF)
PIE	PIE01 PIE02 PIE03 PIE05 PIE06	3332 1629 1632 3329 1632	1024	68	20
				Total	49

[a] The symbol "#" abbreviates "number of".
[b] Office+Caltech10 was represented with two types of features: SURF and DeCAF.

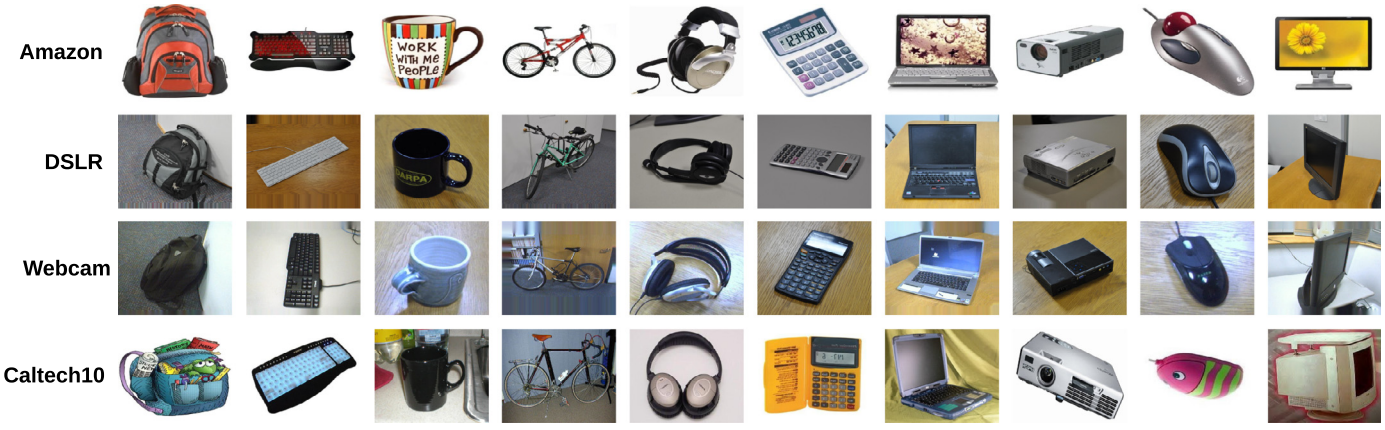

Fig. 3. Qualitative differences between the four domains from the Office+Caltech10 cross-domain set. Amazon: images downloaded from online merchants; DSLR: high-resolution images by a digital SLR camera; Webcam: low-resolution images by a digital web camera; and Caltech10: images downloaded from Google.

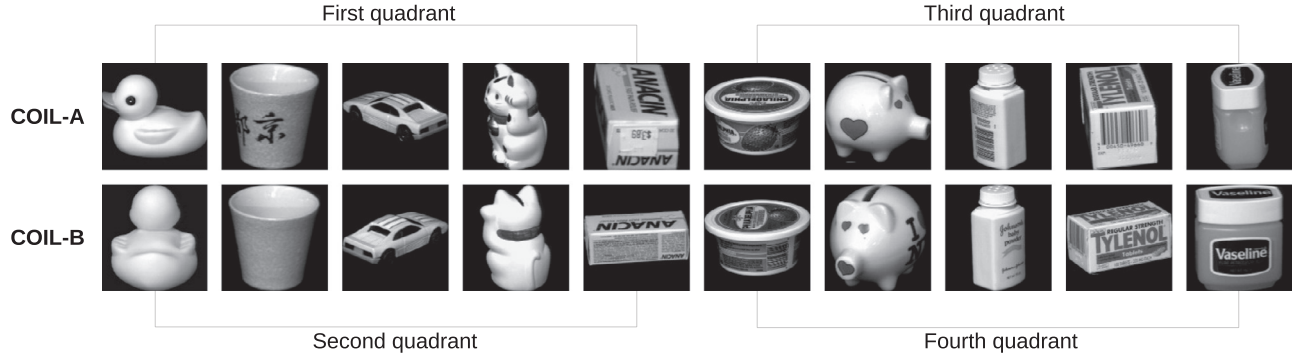

Fig. 4. Qualitative differences between the two domains from the COIL20 dataset. The COIL-A domain (top row) contains images of objects positioned in the first and third quadrant, while the COIL-B domain (bottom row) contains images of objects positioned in the second and fourth quadrant. The differences between the pose of the objects induce a domain shift across the two domains.

datasets [13], what is possibly caused by the differences in the image acquisition methodologies (Fig. 5). From each dataset, subsets (domains) were randomly sampled and two cross-domain problems were constructed using (i) MNIST as source and USPS as target domain; and (ii) USPS as source and MNIST as target domain. All images were rescaled to 16×16 and their gray-scale pixel values were used as image representation.

PIE (**P**ose, **I**llumination, **E**xpression) is a dataset of face images taken from 68 individuals. Each face image was captured by 13 synchronized cameras to get different poses and 21 flashes to get different illuminations and/or expressions [33]. In our experiments, five subsets (domains) were sampled from PIE: PIE-A containing left pose images; PIE-B containing upward pose images; PIE-C containing downward pose images; PIE-D containing frontal pose im-

Fig. 5. Qualitative differences between the two domains from the Digits cross-domain set. Images from the MNIST domain are displayed on the top row, while images from the USPS domains are displayed on the bottom row. The different gray level distributions induce a domain shift across them.

Fig. 6. Qualitative differences between the five domains from the PIE dataset. The PIE-A domain contains left pose images; The PIE-B domain contains upward pose images; The PIE-C domain contains downward pose images; The PIE-D domain contains frontal pose images; and the PIE-E domain contains right pose images. Pose, illumination, and expression differences induce different levels of domain shift across these five domains.

Fig. 7. Qualitative differences between the two domains from the Bing+Caltech256 set. The Bing domain (top row) contains weakly labeled images from the Bing search engine, while the Caltech256 domain (bottom row) contains image download from Google.

ages; and PIE-E containing right pose images (Fig. 6). The differences in pose, lighting, illumination and expression conditions induce different levels of domain shift across these five domains. All combination pairs of them were evaluated as a cross-domain problem using a domain as source and another as target domain. The feature vectors are provided by [21].[7]

Bing+Caltech256 is a cross-domain set composed of two datasets namely, Bing and Caltech256.[8] Bing contains weakly labeled images from the Bing search engine, while Caltech256 contains image downloaded from Google (Fig. 7). In the experiments, Bing was used as source domain whereas Caltech256 as target domain. To simulate different cross-domain scenarios, the number of examples per class in Bing was set to 5 in a first experiment and to 10 in a second experiment. Furthermore, six different number of classes (5, 10, 15, 20, 25, 30) were evaluated in both experiments. Classemes features [2] were used as image representation. This methodology was also used in [13].

Throughout the experiments, only 3 labeled images from target domain were used for semi-supervised domain adaptation purposes.

6.2. Baselines and setup

SSTS was compared with ten baselines, among them 9 state-of-the-art domain adaptation methods:

[7] The five subsets from PIE are available at http://ise.thss.tsinghua.edu.cn/~mlong/doc/joint-distribution-adaptation-iccv13.zip (as of 12/09/2016).

[8] Bing-Caltech256 is available at http://vlg.cs.dartmouth.edu/projects/domainadapt/ (as of 12/09/2016).

GFK (Geodesic Flow Kernel[9]) is an unsupervised domain adaptation method [14]. LDA was used to take advantage of the available discriminative information in source domain, while PCA was used to learn the target domain subspace. This methodology was also used in [13].

HFA (Heterogeneous Feature Augmentation[10]) is a semi-supervised domain adaptation method [20]. Following the authors, the parameter λ was set to 100.

KMA (Kernel Manifold Alignment[11]) is a semi-supervised domain adaptation method [36]. Following the authors, the number of neighbors was set to 21.

mSDA (Marginalized Stacked Denoising Autoencoder[12]) is a fully unsupervised domain adaptation method [5]. Following the authors, the number of layers was set to 5 and the corruption probability was tuned in the range {0.5, 0.6, 0.7, 0.8, 0.9}.

SA (Subspace Alignment[13]) is a fully unsupervised domain adaptation method [12]. LDA was used to take advantage of the available discriminative information in the source domain, while PCA was used to learn the target domain subspace. This methodology was also used in [13].

TCA (Transfer Component Analysis) is a fully unsupervised domain adaptation method [28]. TCA was implemented following the original paper description.

SSTCA (Semi-Supervised Transfer Component Analysis) is a semi-supervised domain adaptation method [28]. SSTCA was also implemented following the original paper description. The number of nearest neighbors was tuned in the range {10, 30, 50, 80, 100}, while the parameter γ was tuned in the range $\{10^{-6}, 10^{-5}, \ldots, 10^{-1}\}$.

TKL (Transfer Kernel Learning[14]) is a kernel-based fully unsupervised domain adaptation method [23]. As suggested by the authors, the RBF kernel was employed, the eigenspectrum damping factor was set to 1.1, and the SVM complexity regularizer was set to 10.

TJM (Transfer Joint Matching[15]) is a fully unsupervised domain adaptation method [22]. Following the authors' suggestion, the regularization parameter was set to 1, while the number of iteration was set to 10.

NA (No Adaptation) is the Nearest Neighbor (NN) classifier trained on source domain and tested on target domain without any domain adaptation. NA served as a control method to measure the effect size of the domain adaptation yielded by SSTS and the baselines.

The subspace dimension for GFK and SA was defined by the number of eigenvectors minus one because LDA was used to project the source subspace. For SSTS, KMA, SSTCA, TCA, and TJM, the dimensionality was tuned in the range $\{10, 20, \ldots, 100\}$. Except for HFA and TKL, which are SVM-based methods, the NN classifier was employed after all domain adaptation method. The classification accuracy served as a quality measure for the domain adaptation, which implies that the higher the classification accuracy, the better the domain adaptation. Note that this experimental protocol is commonly employed in the domain adaptation literature (*e.g.*, [13,14]).

6.3. Statistical analysis

Firstly, the Friedman's test for multiple comparisons was performed considering all sets of cross-domain problems. According to this test, there were significant differences between SSTS and the baseline methods. Then, the Bonferroni-Dunn's post hoc test was performed in order to verify from which baseline SSTS was different in terms of classification accuracy. Regarding the computational load comparison, the Friedman's test was performed considering all cross-domain problems of PIE dataset. There were significant differences between SSTS and the baseline methods. Then, the Bonferroni-Holm's post hoc test was performed. In all tests, significant differences were assumed at $p < 0.05$.

7. Results

In what follows, we present the effect of SSTS on each set of cross-domain problems. For this, we will use the notation *source*↦*target* to denote a cross-domain problem in which the classifier was trained on "*source*" and tested on "*target*" after the domain adaptation.

SSTS increased the classification accuracy in all manually labeled cross-domain problems. Representing Office+Caltech10 with different descriptors did not affect negatively the SSTS performance in increasing the cross-domain classification accuracy. With respect to SURF descriptor, SSTS increased 41% on average (*i.e.*, regarding the twelve problems) the cross-domain classification accuracy in comparison to NA; moreover, it yielded the best classification rates in six out of twelve cross-domain problems (Fig. 8). Regarding the DeCAF descriptor, all methods, including NA, improved their performance in comparison to SURF features, what led to an increasing of classification accuracy in the twelve cross-domain problems (Fig. 9). In this setting, SSTS increased 13% on average the cross-domain classification in comparison to NA; furthermore, it afforded to the best classification rates in nine out of twelve cross-domain problems. Unlike SSTS, the KMA method reduced the classification rates on DSLR↦Webcam in 4% and Webcam↦DSLR in 3% using the DeCAF features. SSTS also increased the cross-domain classification accuracy in the gray-scale problems. The rates were increased in 9% on COIL20 (Fig. 10), 13% on Digits (Fig. 11) and 48% on PIE (Figs. 12 and 13) on average in comparison to NA. Additionally, SSTS afforded the best classification accuracy on all cross-domain problems; the difference in performance between SSTS and the baselines was more apparent on the PIE cross-domain problems (Figs. 12 and 13). Unlike SSTS, all the baselines reduced the classification accuracy in comparison to NA at least once in the gray-scale cross-domain problem (Figs. 10–13).

SSTS increased the classification accuracy on the weakly labeled cross-domain problem. SSTS was less sensitive than the baselines in the experiments with Bing+Caltech256 (Fig. 14). It kept the classification accuracy higher than NA even when the number of classes increases. This behavior was independent of the instance number per class: on average, SSTS yielded 45.8% and 47.2% of accuracy in the experiments with five and ten samples per class respectively. These rates are 12% and 11% more accurate than the ones yielded by NA and 4.5% and 7.0% more accurate than the ones yielded by HFA, the best baseline. Unlike SSTS, the baselines reduced the classification accuracy at least one time in the experiments. This behavior was observed in both experimental conditions, the variation of instance number per class and the number class increase (Fig. 14).

[9] GFK source code is available at http://www-scf.usc.edu/~boqinggo/domain_adaptation/GFK_v1.zip (as of 12/09/2016).

[10] HFA source code is available at http://www.lxduan.info/#sourcecode_hfa (as of 12/09/2016).

[11] KMA source code is available at https://github.com/dtuia/KEMA.git (as of 12/09/2016).

[12] mSDA source code is available at http://www.cse.wustl.edu/~mchen/papers/deepmsda.eps (as of 12/09/2016).

[13] SA source code is available at http://users.cecs.anu.edu.au/~basura/DA_SA/DA_SA.zip (as of 12/09/2016).

[14] TKL source code is available at http://ise.thss.tsinghua.edu.cn/~mlong/doc/transfer-kernel-learning-tkde15.zip (as of 12/09/2016).

[15] TJM source code is available at http://ise.thss.tsinghua.edu.cn/~mlong/doc/transfer-joint-matching-cvpr14.zip (as of 12/09/2016).

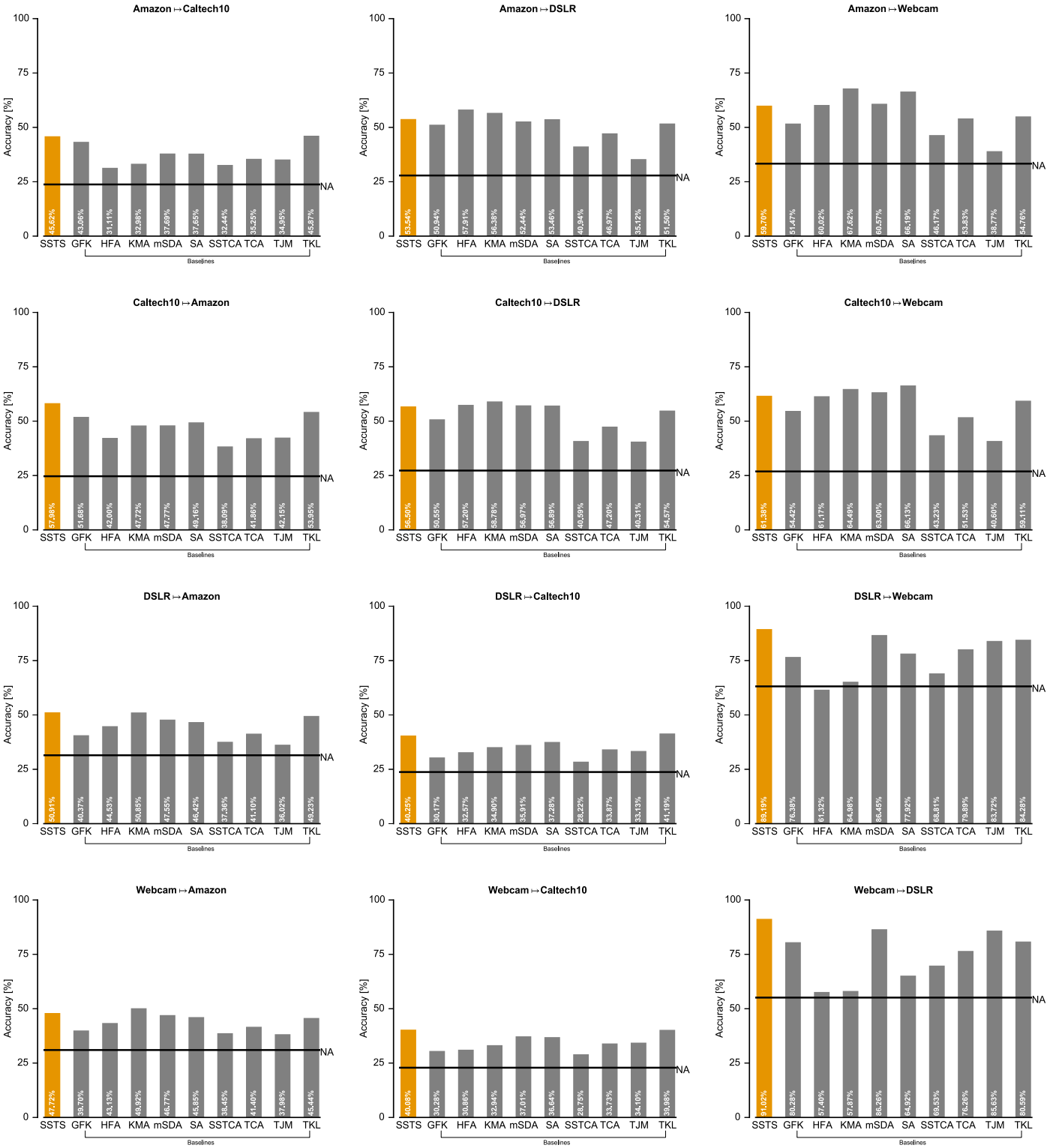

Fig. 8. Effect of SSTS on the classification of Office+Caltech10 cross-domain problems represented with SURF descriptor. Each cross-domain problem is titled as source↦target. No adaptation (NA) accuracy is depicted as the black horizontal bars. The accuracy of each method is inside its respective bar.

Overall, SSTS afforded the best cross-domain classification accuracy in all experimental conditions, *i.e.*, considering all the 49 cross-domain problems (Fig. 15).

Computational load and parameter choice. SSTS has an acceptable execution time (Fig. 16). It was not as fast as mSDA, SA, or TKL; however, SSTS took on average the same computational load as GFK, SSTCA, and TCA. In comparison to the slowest methods, SSTS was around 8 times faster than TJM, 20 times faster than KMA, and more than 1000 times faster than HFA. Regarding to the hyper-parameters (Fig. 17), SSTS performed better with small values of $\beta \in [10^{-6}, 10^{-4}]$ and with $\gamma \in [10^{-3}, 1]$. Furthermore, low

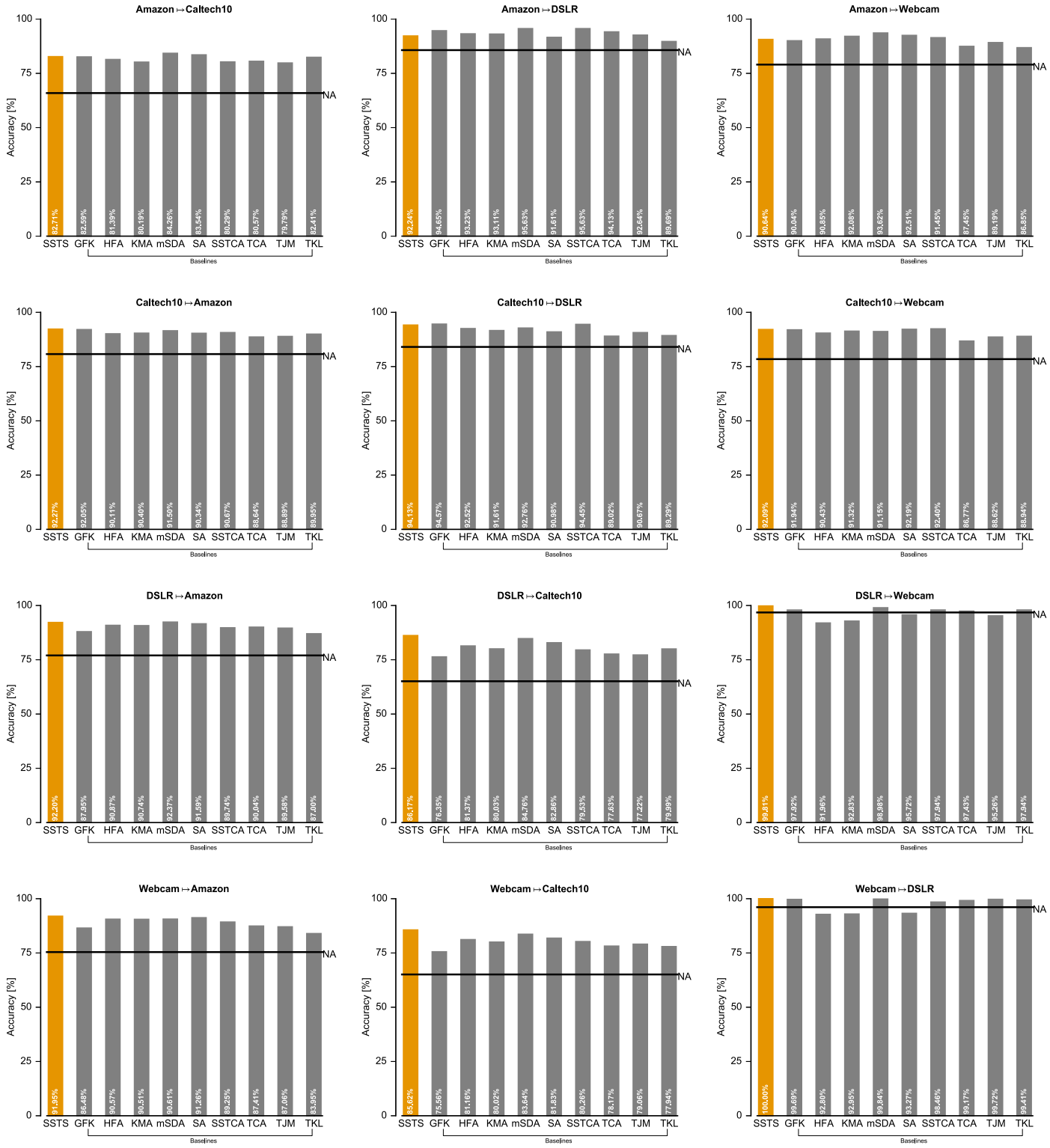

Fig. 9. Effect of SSTS on the classification of Office+Caltech10 cross-domain problems represented with DeCAF descriptor. Each cross-domain problem is titled as source↦target. No adaptation (NA) accuracy is depicted as the black horizontal bars. The accuracy of each method is inside its respective bar.

dimensional subspaces (with $d \in [10, 20, \ldots, 100]$) were suitable to improve the classification accuracy.

8. Discussion

Here we presented SSTS, a new semi-supervised domain adaptation method. SSTS reduces significantly the domain shift in cross-domain problems. It is equipped with a simple and effective mechanism to reduce the domain shift based on two processes: (i) constraining pairs of instances across domains; and (ii) exploiting global statistical properties of unlabeled target data. While the first process explores the discriminative knowledge from source and target labeled data, the second explores the unlabeled target data – always in abundance. By encoding these two processes in

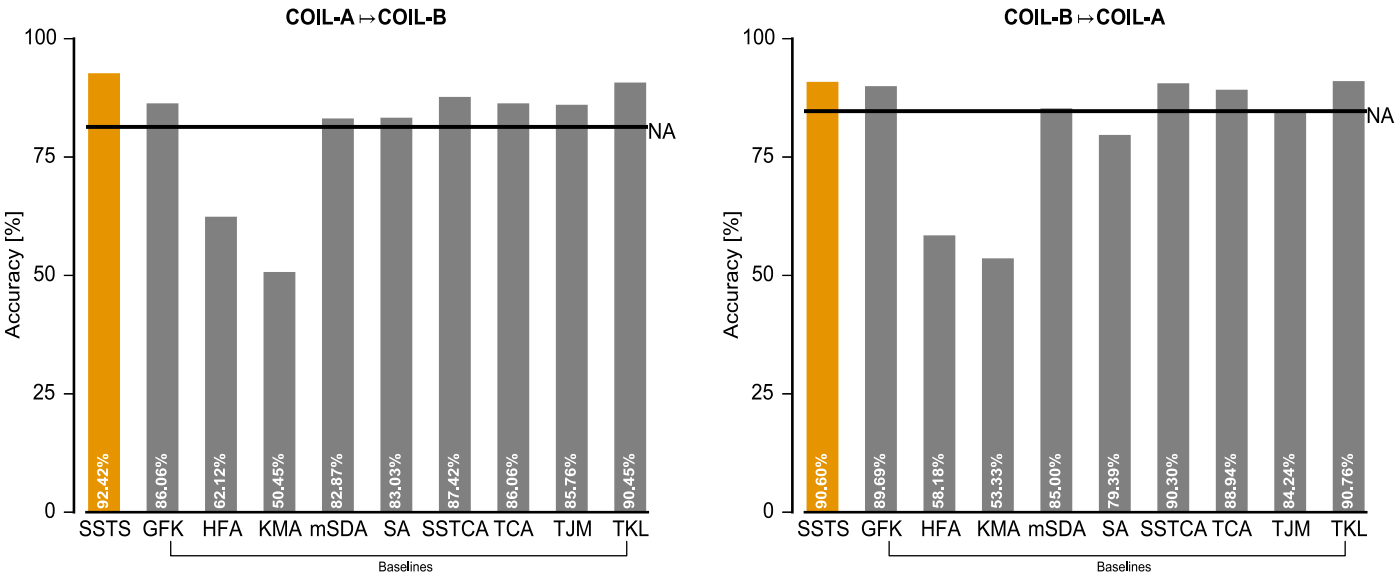

Fig. 10. Effect of SSTS on the classification of COIL20 cross-domain problems. Each cross-domain problem is titled as source↦target. No adaptation (NA) accuracy is depicted as the black horizontal bars. The accuracy of each method is inside its respective bar.

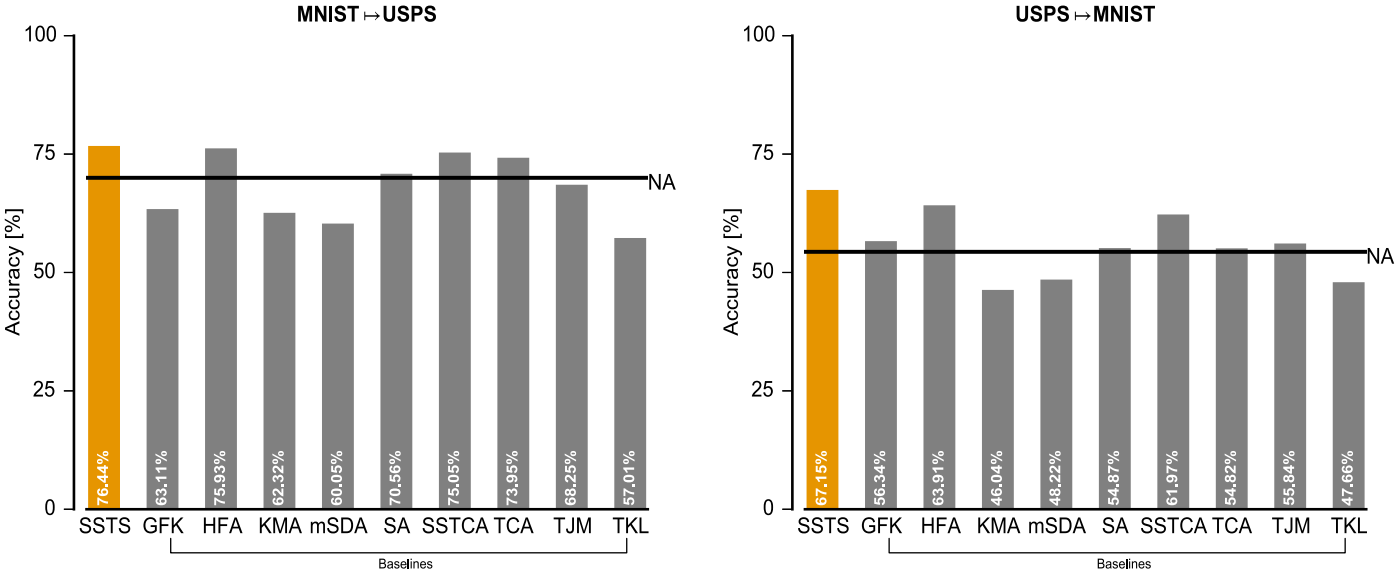

Fig. 11. Effect of SSTS on the classification of Digits cross-domain problems. Each cross-domain problem is titled as source↦target. No adaptation (NA) accuracy is depicted as the black horizontal bars. The accuracy of each method is inside its respective bar.

a (non)linear transformation, SSTS builds a domain invariant subspace (*i.e.*, an implicit metric) where the difference between the source and target distributions is reduced. After projecting both domains onto this domain invariant subspace, any classifier trained on the source domain can be used to classify target instances.

A direct consequence of domain shift reduction in cross-domain problems is the classification accuracy increase [13,19,21,36]. Based on this implication, we found that SSTS increased the classification accuracy on 48 manually-labeled cross-domain problems. Therefore, SSTS reduced the domain shift in all of these problems. This finding is important because each problem has difference domain shift level (Figs. 3–7); and performing well on variety condition is a desired property of any domain adaptation method. Recent studies, however, show that some subspace-based methods reduce the domain shift of Office+Caltech10 when represented with SURF but do not when represented with DeCAF [19,36]. Nevertheless, this was not observed with SSTS; the results were the expected: as DeCAF provided richer features than SURF [10], the classification accuracies were higher than the ones observed with SURF features. Moreover, SSTS performed the best on 37 out of 48 problems and on paired with the baselines on the remaining ones, what shows that SSTS is more effectively in reducing the domain shift than the state-of-the-art methods.

Besides being effective in reducing domain shift in presence of manually labeled data, SSTS also reduces domain shift in presence of weakly labeled data. Applying domain adaptation on such data, however, is challenging because the label associated with each source domain instance may be unreliable [2]. Due to this drawback, it is reasonable to expect that increasing weakly labeled data and/or increasing the number of classes may harm the domain adaptation and consequently reduce the cross-domain classification accuracy. This effect was indeed observed in our experiments; nevertheless, SSTS was less affected than the baselines (Fig. 14). Despite the decrease in accuracy (1% on average) due to the in-

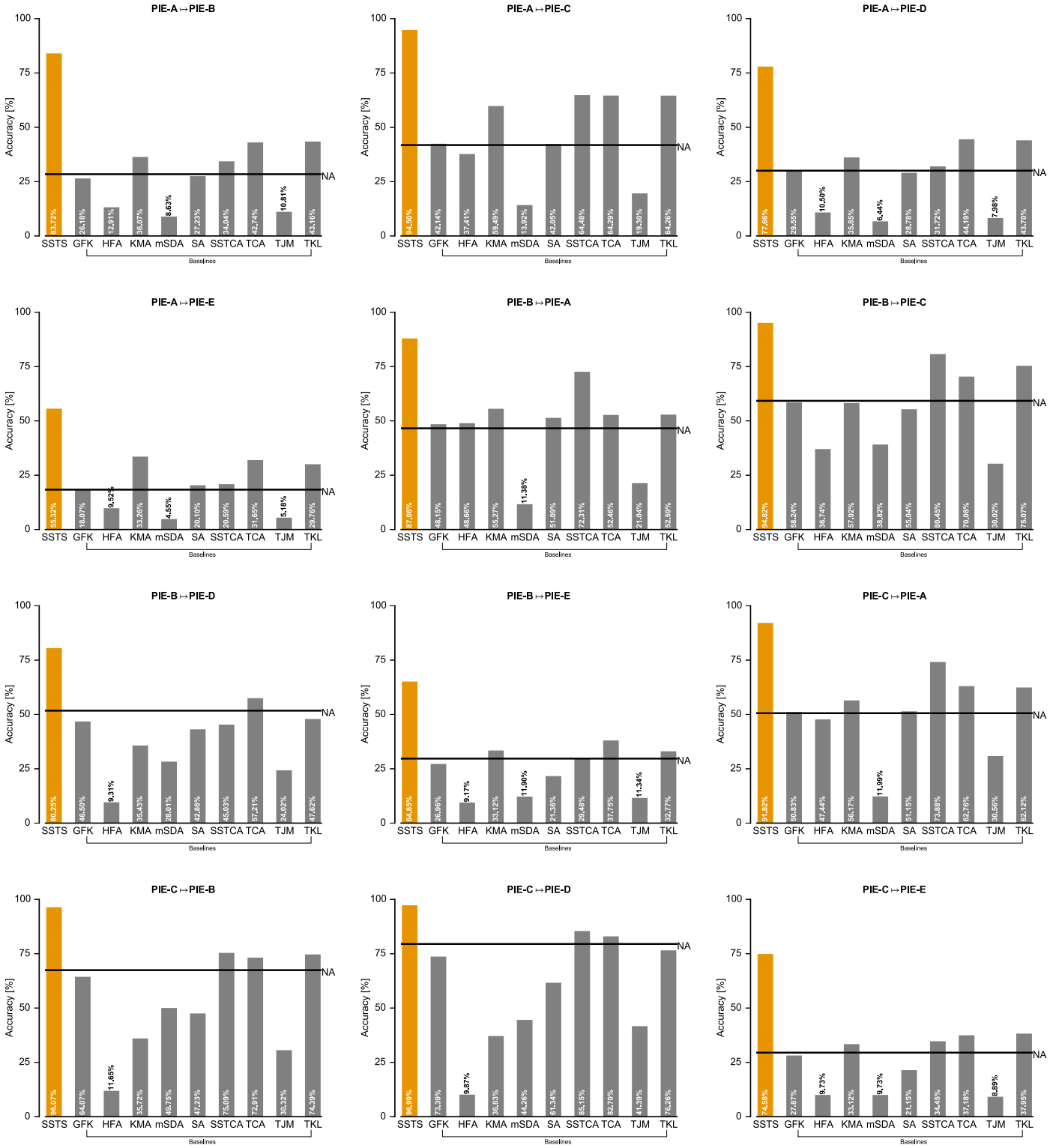

Fig. 12. Effect of SSTS on the classification of PIE cross-domain problems. Each cross-domain problem is titled source↦target. No adaptation (NA) accuracy is depicted as the black horizontal bars. The accuracy of each method is inside its respective bar. The remaining results on PIE are displayed in Fig. 13.

crease of weakly labeled data, SSTS reduced the domain shift leading to an accuracy better than NA (11% on average). In contrast to HFA, the best baseline, this result was 6.8% better. Moreover, SSTS and HFA were the only methods that kept reducing the domain shift with the addition of new classes. These results support that SSTS can handle the complexity imposed by weakly labeled data.

Another important SSTS' property is the capacity of avoiding negative domain transfer (NDT). NDT occurs when the domain adaptation algorithm increases the domain shift instead of reducing it, thereby decreasing the classification accuracy [27]. Despite being a relevant aspect when designing a domain adaptation method, NDT does not receive much importance (*e.g.* [13,19,36]). In presence of manually or weakly labeled data, SSTS was robust

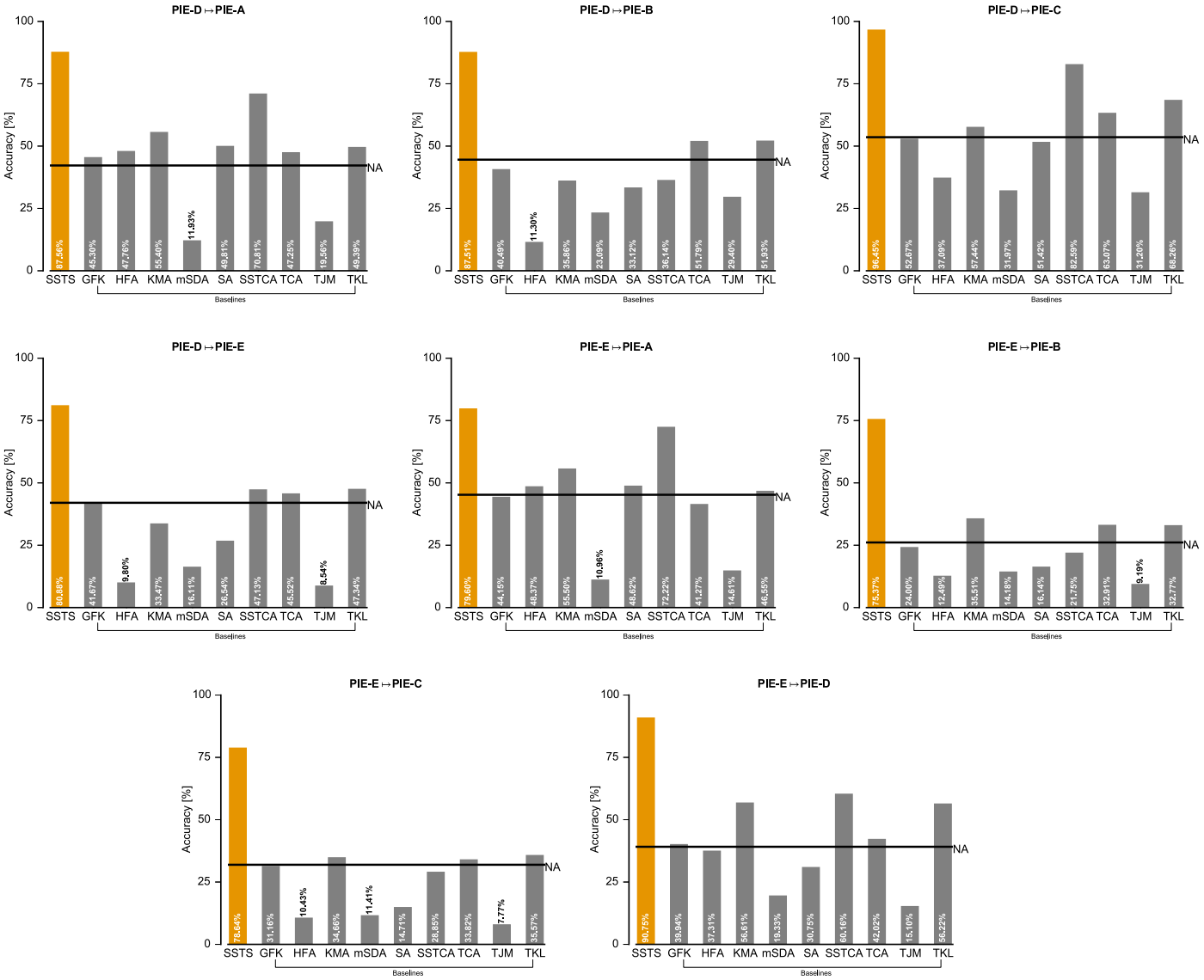

Fig. 13. Effect of SSTS on the classification of PIE cross-domain problems. Each cross-domain problem is titled source↦target. No adaptation (NA) accuracy is depicted as the black horizontal bars. The accuracy of each method is inside its respective bar. The remaining results on PIE are displayed in Fig. 12.

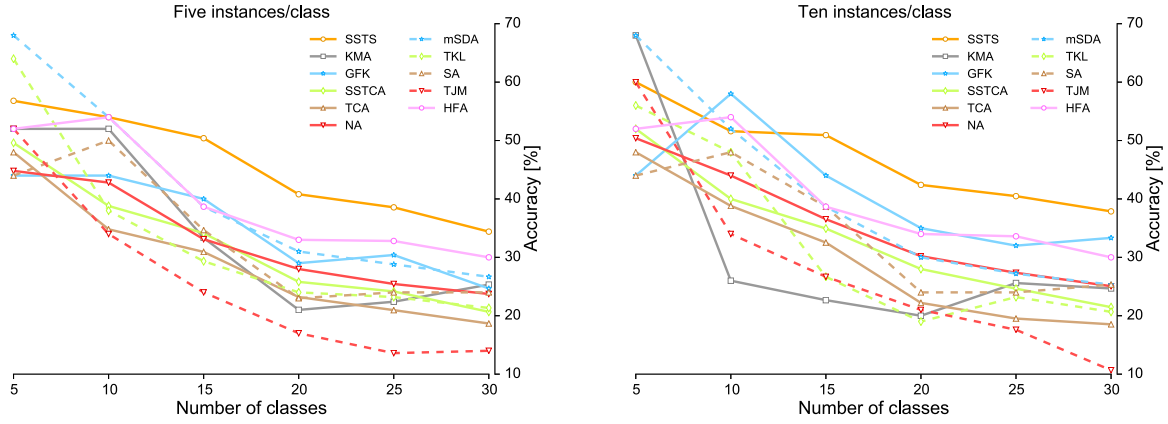

Fig. 14. SSTS' sensitivity on Bring+Caltech256 regarding the instances per class and the number of classes. The Bing dataset was employed as the source while the Caltech256 dataset as target domain. The left chart displays the experimental results using 5 instances per class, whereas the right chart displays the experimental results using 10 instances per class.

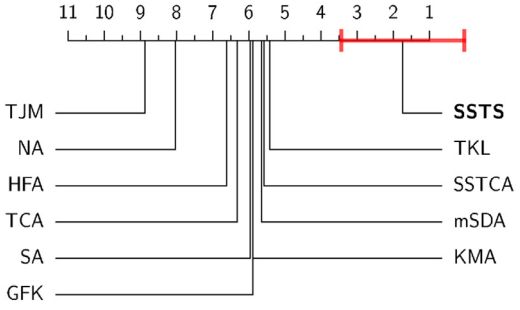

Fig. 15. The Critical Distance diagram of *SSTS vs.* the baselines. The better a method is ranked, the higher is the average performance (on all cross-domain problems) it achieves. Methods ranked outside the marked interval (red colored) were considered different from SSTS, which implies all the baselines. Statistics: Friedman's test ($\chi = 171.1$; $p < 0.000001$) followed by Bonferroni-Dunn post hoc test ($p < 0.05$). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

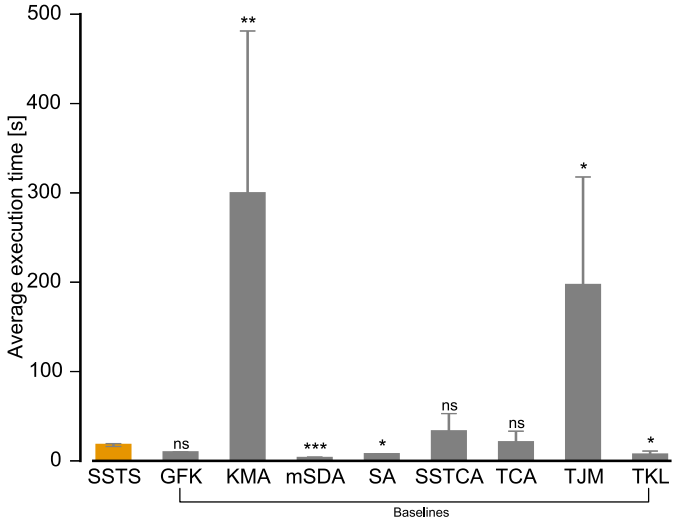

Fig. 16. SSTS execution time on PIE dataset. The experiments were performed on a workstation with Intel Xeon 2.93 GHz and 12GB of RAM. HFA took more than 2 days, thus we did not include it to enhance the visualization of the other results. Statistics: Friedman's test ($\chi = 170.3$; $p < 0.000001$) followed by Bonferroni-Holm post hoc test ("ns" stands for $p > 0.05$; $^{*}p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$).

against NDT (Figs. 8–14). Even on the Digits problems, which has been reported as a challenging domain adaptation problem [13], SSTS reduced the domain shift (Fig. 11). The baseline methods, however, were prone to cause NDT. For instance, KMA failed to reduce the domain shift in 11 out of 49 problems, including the easiest problems whose domain shift is only induced by different image capturing devices (DSLR↦Webcam and Webcam↦DSLR in Fig. 9). This aspect corroborates that SSTS reduces effectively the domain shift in cross-domain problems.

Performing domain adaptation with SSTS does not require a high computational load. SSTS takes time similar to state-of-the-art methods, such as GFK, TCA, and SSTCA; and it is less time-consuming than TJM, KMA, and HFA (Fig. 16). Moreover, the non-linear version can be useful to handle data represented in an extremely high dimensional spaces (*i.e.*, $\mathcal{D} \gg N$) [23,26,36]. Additionally, SSTS can make the classification process faster since good performances are achieved by projecting data onto low-dimensional invariant subspaces (Fig. 17).

In certain problems, performing domain adaptation from a unique source domain may be not effective, mainly when such a domain has a few amount of labeled instances. An option, in these cases, is to use more than one source domain, what is called multiple domain adaptation (*e.g.*, [34]). Applying this strategy, however, is not trivial because the multiple sources may have domain shift between themselves. Based on the good performance of SSTS, it looks appealing extend it to the multiple domain adaptation scenarios in a future work.

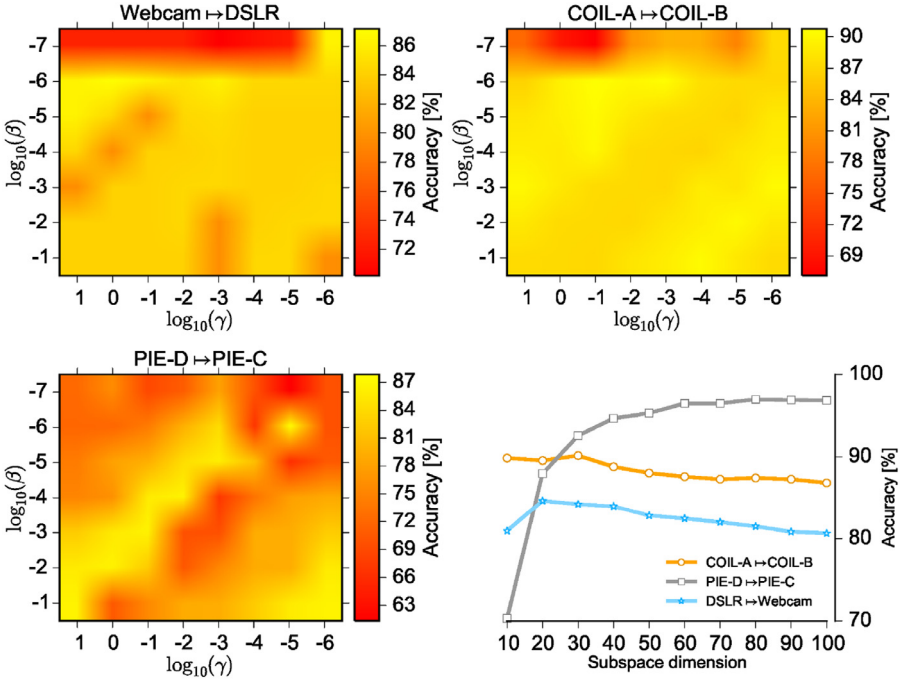

Fig. 17. SSTS parameter analysis. The top (Webcam↦DSLR, COIL-A↦COIL-B) and the bottom-left (PIE-D↦PIE-C) figures display heat maps of the effect of the hyper-parameter $\gamma \in [10^{-6}, 10^{-5}, \ldots, 10]$ and $\beta \in [10^{-7}, 10^{-6}, \ldots, 1]$. The bottom-right figure displays the effect of subspace dimension on the same cross-domain problems.

Acknowledgments

Authors are grateful to CAPES, CNPq, FAPESP (grant 2015/09169-3), and the FAPESP-Microsoft Virtual Institute (grants 2013/50169-1 and 2013/50155-0).

References

[1] Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1798–1828.
[2] A. Bergamo, L. Torresani, Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach, in: Advances in Neural Information Processing Systems, 2010, pp. 181–189.
[3] J. Blitzer, R. McDonald, F. Pereira, Domain adaptation with structural correspondence learning, in: Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2006, pp. 120–128.
[4] L. Bruzzone, M. Marconcini, Domain adaptation problems: a dasvm classification technique and a circular validation strategy, IEEE Trans. Pattern Anal. Mach. Intell. 32 (5) (2010) 770–787.
[5] M. Chen, Z. Xu, K. Weinberger, F. Sha, Marginalized denoising autoencoders for domain adaptation, in: Proceedings of the 29th International Conference on Machine Learning, ACM, 2012, pp. 767–774.
[6] L. Cheng, S.J. Pan, Semi-supervised domain adaptation on manifolds, IEEE Trans. Neural Netw. Learn. Syst. 25 (12) (2014) 2240–2249.
[7] F.R.K. Chung, Spectral Graph Theory, vol. 92, American Mathematical Society, 1997.
[8] T. De Bie, N. Cristianini, R. Rosipal, Eigenproblems in pattern recognition, in: Handbook of Geometric Computing, Springer, 2005, pp. 129–167.
[9] J. Deng, Z. Zhang, F. Eyben, B. Schuller, Autoencoder-based unsupervised domain adaptation for speech emotion recognition, IEEE Signal Process. Lett. 21 (9) (2014) 1068–1072.
[10] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: a deep convolutional activation feature for generic visual recognition, in: Proceedings of The 31st International Conference on Machine Learning, 2014, pp. 647–655.
[11] L. Duan, D. Xu, I. Tsang, J. Luo, Visual event recognition in videos by learning from web data, IEEE Trans. Pattern Anal. Mach. Intell. 34 (9) (2012) 1667–1680.
[12] B. Fernando, A. Habrard, M. Sebban, T. Tuytelaars, Unsupervised visual domain adaptation using subspace alignment, in: Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 2960–2967.
[13] B. Fernando, T. Tommasi, T. Tuytelaars, Joint cross-domain classification and subspace learning for unsupervised adaptation, Pattern Recognit. Lett. 65 (2015) 60–66.
[14] B. Gong, Y. Shi, F. Sha, K. Grauman, Geodesic flow kernel for unsupervised domain adaptation, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 2066–2073.
[15] R. Gopalan, R. Li, R. Chellappa, Domain adaptation for object recognition: an unsupervised approach, in: International Conference on Computer Vision, IEEE, 2011, pp. 999–1006.
[16] R. Gopalan, R. Li, V. Patel, R. Chellappa, Domain adaptation for visual recognition, Found. Trends Comput. Graphics Vision 8 (4) (2015) 285–378.
[17] X. He, S. Yan, Y. Hu, P. Niyogi, H. Zhang, Face recognition using laplacianfaces, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 328–340.
[18] A. Khosla, T. Zhou, T. Malisiewicz, A.A. Efros, A. Torralba, Undoing the damage of dataset bias, in: European Conference on Computer Vision, Springer, 2012, pp. 158–171.
[19] W.M. Kouw, L.J.P. van der Maaten, J.H. Krijthe, M. Loog, Feature-level domain adaptation, J. Mach. Learn. Res. 17 (171) (2016) 1–32.
[20] W. Li, L. Duan, D. Xu, I.W. Tsang, Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation, IEEE Trans. Pattern Anal. Mach. Intell. 36 (6) (2014) 1134–1148.
[21] M. Long, J. Wang, G. Ding, J. Sun, P.S. Yu, Transfer feature learning with joint distribution adaptation, in: International Conference on Computer Vision, IEEE, 2013, pp. 2200–2207.
[22] M. Long, J. Wang, G. Ding, J. Sun, P.S. Yu, Transfer joint matching for unsupervised domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1410–1417.
[23] M. Long, J. Wang, J. Sun, S.Y. Philip, Domain invariant transfer kernel learning, IEEE Trans. Knowl. Data Eng. 27 (6) (2015) 1519–1532.
[24] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, K.R. Mullers, Fisher discriminant analysis with kernels, in: Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop., 1999, pp. 41–48.
[25] S.A. Nene, S.K. Nayar, H. Murase, et al., Columbia Object Image Library (COIL-20), Technical Report, CUCS-006-96, Columbia University, New York, N.Y. 10027, 1996.
[26] S. Pan, I. Tsang, J. Kwok, Q. Yang, Domain adaptation via transfer component analysis, IEEE Trans. Neural Networks 22 (2) (2011) 199–210.
[27] S. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[28] S.J. Pan, I.W. Tsang, J.T. Kwok, Q. Yang, Domain adaptation via transfer component analysis, IEEE Trans. Neural Networks 22 (2) (2011) 199–210.
[29] V.M. Patel, R. Gopalan, R. Li, R. Chellappa, Visual domain adaptation: a survey of recent advances, IEEE Signal Process. Mag. 32 (3) (2015) 53–69.
[30] K. Saenko, B. Kulis, M. Fritz, T. Darrell, Adapting visual category models to new domains, in: European Conference on Computer Vision, Springer, 2010, pp. 213–226.
[31] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput. 10 (5) (1998) 1299–1319.
[32] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2002.
[33] T. Sim, S. Baker, M. Bsat, The cmu pose, illumination, and expression (pie) database, in: International Conference on Automatic Face and Gesture Recognition, IEEE, 2002, pp. 46–51.
[34] S. Sun, H. Shi, Y. Wu, A survey of multi-source domain adaptation, Inf. Fusion 24 (2015) 84–92.
[35] A. Torralba, A. Efros, Unbiased look at dataset bias, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2011, pp. 1521–1528.
[36] D. Tuia, G. Camps-Valls, Kernel manifold alignment for domain adaptation., PLoS ONE 11 (2) (2016).
[37] U. Von Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (4) (2007) 395–416.
[38] C. Wang, S. Mahadevan, Heterogeneous domain adaptation using manifold alignment, in: Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Two, AAAI Press, 2011, pp. 1541–1546.
[39] M. Xiao, Y. Guo, Feature space independent semi-supervised domain adaptation via kernel matching, IEEE Trans. Pattern Anal. Mach. Intell. 37 (1) (2015) 54–66.
[40] T. Yao, Y. Pan, C. Ngo, H. Li, T. Mei, Semi-supervised domain adaptation with subspace learning for visual recognition, in: Conference on Computer Vision and Pattern Recognition, 2015, pp. 2142–2150.
[41] R.S. Zemel, M.A. Carreira-Perpinán, Proximity graphs for clustering and manifold learning, in: Advances in Neural Information Processing Systems, 2004, pp. 225–232.

Luis Pereira received the B.Sc. Degree in Information Systems from the São Paulo State University, São Paulo, Brazil, in 2011, and the M.Sc. Degree in Computer Science from the same university, in 2013. Currently, he is pursuing the Ph.D. degree in Computer Science at University of Campinas with the Institute of Computing. His research focuses on the areas of machine learning, computer vision, graph-based algorithms, and transfer learning, with a particular interest in domain adaptation systems.

Ricardo da Silva Torres is Full Professor of computer science at the University of Campinas (UNICAMP). Dr. Torres received a B.Sc. in Computer Engineering from University of Campinas, Brazil, in 2000 and his Ph.D. degree in Computer Science at the same university in 2004. Dr. Torres is co-founder and member of the RECOD lab, where he has been developing multidisciplinary e-Science research involving Multimedia Analysis, Multimedia Image Retrieval, Databases, Digital Libraries, and Geographic Information Systems. Dr. Torres is author/co-author of more than 100 articles in refereed journal and conferences and serves as PC member for several international and national conferences.