Int J Comput Vis (2014) 109:28–41
DOI 10.1007/s11263-014-0719-3

Asymmetric and Category Invariant Feature Transformations for Domain Adaptation

Judy Hoffman · Erik Rodner · Jeff Donahue · Brian Kulis · Kate Saenko

Received: 21 March 2013 / Accepted: 17 March 2014 / Published online: 13 April 2014

Abstract We address the problem of visual domain adaptation for transferring object models from one dataset or visual domain to another. We introduce a unified flexible model for both supervised and semi-supervised learning that allows us to learn transformations between domains. Additionally, we present two instantiations of the model, one for general feature adaptation/alignment, and one specifically designed for classification. First, we show how to extend metric learning methods for domain adaptation, allowing for learning metrics independent of the domain shift and the final classifier used. Furthermore, we go beyond classical metric learning by extending the method to asymmetric, category independent transformations. Our framework can adapt features even when the target domain does not have any labeled examples for some categories, and when the target and source features have different dimensions. Finally, we develop a joint learning framework for adaptive classifiers, which outperforms competing methods in terms of multi-class accuracy and scalability. We demonstrate the ability of our approach to adapt object recognition models under a variety of situations, such as differing imaging conditions, feature types, and codebooks. The experiments show its strong performance compared to previous approaches and its applicability to large-scale scenarios.

Keywords Object recognition · Domain adaptation · Transformation learning

Communicated by Hal Daumé.

J. Hoffman (✉) · J. Donahue
UC Berkeley, Berkeley, CA, USA
e-mail: jhoffman@eecs.berkeley.edu

J. Donahue
e-mail: jdonahue@cs.berkeley.edu

E. Rodner
Friedrich Schiller University of Jena, Jena, Germany
e-mail: erik.rodner@uni-jena.de

B. Kulis
Ohio State University, Columbus, OH, USA
e-mail: kulis@cse.ohio-state.edu

K. Saenko
University of Massachusetts, Lowell, MA, USA
e-mail: saenko@cs.uml.edu

1 Introduction

In many real-world applications of object recognition, the image distribution used for training (source dataset, or *domain*) is different from the image distribution used for testing (target domain). This distribution shift is typically caused by data collection bias (see Fig. 1 for three example domains collected for the same set of object categories.) In general, visual domains can differ in a combination of (often unknown) factors, including scene, intra-category variation, object location and pose, viewing angle, resolution, motion blur, scene illumination, background clutter, camera characteristics, etc. Recent studies have demonstrated a significant degradation in the performance of state-of-the-art image classifiers due to domain shift from pose changes (Farhadi and Tabrizi 2008), a shift from commercial to consumer video (Duan et al. 2009, 2012b), and, more generally, training datasets biased by the way in which they were collected (Torralba and Efros 2011).

Methods for *adapting* to a target distribution have been proposed, both in and outside of the vision community. Some have focused on learning adapted classifier parameters, typically by minimizing classification error using a small number of (category) labels in the target domain

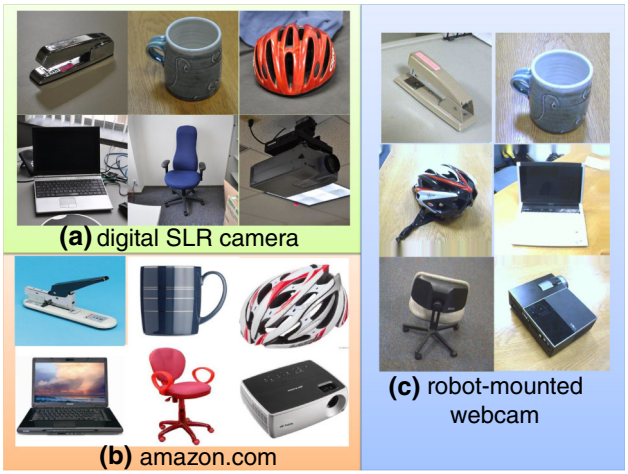


Fig. 1 We address the problem of transfering object category models between visual domains, such as (**a**) high-resolution DSLR photographs of objects taken by a human, (**b**) images downloaded from amazon.com, and (**c**) images captured by a robot-mounted webcam. Each domain is characterized by a distinct feature distribution caused by, e.g., background clutter in (**a**, **c**) versus uniform backgrounds in (b), or fine-grained detail in (**a**, **b**) versus lack thereof in (**c**) (as on the laptop keyboard)

(Bergamo and Torresani 2010; Jiang et al. 2008; Yang et al. 2007, etc.). Others use existing classifiers but learn a transformation between the *features* in the various domains, either by utilizing unlabeled but corresponding points across domains, such as a scene captured simultaneously from multiple views (Farhadi and Tabrizi 2008), or by somehow aligning the domain distributions (Gopalan et al. 2011; Gong et al. 2012).

In this paper, we introduce a novel domain adaptation technique that improves on and combines the above approaches. We first propose a novel method of learning a domain-invariant feature space, and later extend this formulation to simultaneously adjust the decision boundary *in the new space*, using all available labeled data.

The key idea behind our feature adaptation method is to learn a *regularized transformation* that maps feature points from one domain to another using cross-domain constraints. The constraints are formed by requiring that the transformation maps points from the same category (but different domain) near each other. Its advantages over previous feature adaptation methods are that (1) it can learn from category labels, and not just from instance-level constraints, (2) it can adapt models between heterogeneous spaces, including those with different dimensions, via an *asymmetric transform*, and (3) the learned transformation is category independent and thus transferrable to unlabeled categories. This method, which we call the **Asymmetric Regularized Cross-domain Transform (arc-t)**, is independent of the classifier and allows us to encode domain invariance into the feature representation of a broad range of classification methods, from k-NN to SVM, as well as clustering methods.

Forcing all intra-class points to be similar can be inefficient when the end goal is to learn a decision boundary. We extend the above to simultaneously learn the transformation of features and the *classifier parameters* themselves, using the same classification loss to jointly optimize both. This method, referred to as **Maximum Margin Domain Transform (mmdt)**, provides a way to adapt max-margin classifiers in a multi-class manner, by learning a shared component of the domain shift as captured by the feature transformation.

Because it operates over the input features, **mmdt** can generalize the learned shift in a way that parameter-based methods cannot. On the other hand, it overcomes the limitations of the **arc-t** method as applied to classification: by optimizing the classification loss directly in the transform learning framework, it can achieve higher accuracy; furthermore, its use of efficient hyperplane constraints significantly reduces the training time of the algorithm, and learning a transformation directly from target to source allows efficient optimization in linear space.

The article builds on several conference publications. The transform learning methods for domain adaptation have been presented in Saenko et al. (2010) (for symmetric metrics) and in Kulis et al. (2011) (asymmetric **arc-t**). The max-margin formulation was introduced in Hoffman et al. (2013). This work presents a unified framework for all three methods, and further insights into their underlying connections. In addition, we present a comprehensive comparison of the methods to each other, as well as to recent visual domain adaptation approaches.

2 Related Work

Domain adaptation, or covariate shift, is a fundamental problem in machine learning and in related fields. It has attracted a lot of attention in the natural language community (Blitzer et al. 2007; Daume III 2007; Ben-david et al. 2007; Jiang and Zhai 2007, etc.) and in computer vision (Bergamo and Torresani 2010; Li and Zickler 2012; Jhuo et al. 2012; Gong et al. 2012, etc.).

The problem statement of domain adaptation is related to multi-task learning, but differs from it in the following way: in domain adaptation problems, the distribution $p(\boldsymbol{x})$ over the features varies across tasks (domains), while the output labels y remain the same; in multi-task learning or knowledge transfer, $p(\boldsymbol{x})$ stays the same across tasks (single domain), while the output labels vary (see Jiang 2008 for more details). In this article, we address *multi-task learning across domains*; i.e., both $p(\boldsymbol{x})$ and the output labels y can change between domains.

In the following, we briefly review domain adaptation methods that either focus on computer vision applications or that are related to our approach. We present a detailed comparison to several of these methods in Sect. 7. A

comprehensive overview of multi-task learning and domain adaptation is given in Jiang (2008).

Classifier adaptation Several classifier-centric approaches have been presented for domain adaptation, most based on the SVM. For example, Bergamo and Torresani (2010) propose a weighted combination of source, target, and transductive SVMs. One of the prominent approaches is given by Daume III (2007), who introduces a feature replication method for domain adaptation. The basic idea is to define augmented feature vectors $\boldsymbol{x}' = (\boldsymbol{x}; \boldsymbol{x}; \boldsymbol{0})$ for data points $\boldsymbol{x}$ in the source and $\tilde{\boldsymbol{x}}' = (\tilde{\boldsymbol{x}}; \boldsymbol{0}; \tilde{\boldsymbol{x}})$ for data points $\tilde{\boldsymbol{x}}$ in the target. Daume also gives an overview of the relevant baselines, which we employ in this work. In the linear case, feature replication (Daume III 2007) can be shown to decompose the learned hyperplane parameter into $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}} + \boldsymbol{\theta}'$, where $\hat{\boldsymbol{\theta}}$ is shared by all domains (Jiang et al. 2008). This is similar to Adaptive SVMs (Yang et al. 2007; Li 2007), where the target classifier $f^T(\boldsymbol{x})$ is adapted from the existing, auxiliary classifier $f^A(\boldsymbol{x})$ via the equation $f^T(\boldsymbol{x}) = f^A(\boldsymbol{x}) + \delta f(\boldsymbol{x})$, where $\delta f(\boldsymbol{x})$ is the perturbation function. The PMT-SVM method of Aytar and Zisserman (2011) is related but uses a different regularization term that does not indirectly penalize the margin.

Domain transfer SVM (Duan et al. 2009) attempts to reduce the mismatch in the domain distributions, measured by the maximum mean discrepancy, while also learning a target decision function. A related method (Duan et al. 2012b) utilizes adaptive multiple kernel learning to learn a kernel function based on multiple base kernels.

The disadvantage of methods that only adapt the classifier is their inability to transfer the learned domain shift to novel categories, which is limiting in object recognition scenarios, where the set of available category labels varies among datasets.

Multi-view learning Multi-view learning (Sharma et al. 2012; Quadrianto and Lampert 2011; Farquhar et al. 2005; Diethe et al. 2010, etc.) addresses the scenario where multiple sets of observations are available per labeled example, resulting in multiple *views* of the data. The views could come from different modalities or, in vision, different 3D pose of the same object instance. For visual domain adaptation, such methods have been applied in cases where multiple observations of the same instance are available (Farhadi and Tabrizi 2008; Dai et al. 2008; Li and Zickler 2012). For example, Kan et al. (2012) use multi-view learning on multiple views of the same face to perform face recognition across pose variation. In contrast to classifier adaptation described above, such methods attempt to learn a perturbation over the feature space, rather than a class-specific perturbation of the model parameters, typically in the form of a transformation matrix or modified kernel. In particular, Farhadi and Tabrizi (2008) as well as Li and Zickler (2012) translate features between camera views to transfer activity models, while Dai et al. (2008) translated between text and image domains.

Our method can handle multiple views, i.e. data with *instance constraints*, when available; however, unlike multi-view learning, it can also handle the case of multiple object category datasets that have no instances in common, and only share the same category labels. To apply multi-view methods in this scenario would require summarizing all instances of a particular category (for example, the mean) and considering the per category feature from the source and target to be multiple views of the same category. For completeness, we did experiments with one such method (Sharma et al. 2012), but found that performance was not comparable to methods created for the category adaptation problem.

Hybrid supervised methods Instead of choosing either feature transformation or classifier adaptation, it is possible to combine the two approaches, as we do in this paper with **mmdt**. The approach most closely related to ours is the recent Heterogeneous Feature Augmentation (HFA) method (Duan et al. 2012a), which learns a feature transformation into a common latent space, as well as the classifier parameters. However, in contrast to **mmdt**, **hfa** is formulated to solve a binary problem, so a new feature transformation must be learned for each category. Therefore, unlike **mmdt**, **hfa** cannot learn a representation that generalizes to novel target categories. Furthermore, our method has better computational complexity.

Semi- and Un-supervised methods While we do not discuss in detail it here, in Donahue et al. (2013) we presented a semi-supervised extension of our model, adding constraints between unlabeled target points to the labeled constraints. For example, we placed smoothness constraints on examples that lay on a consistent motion path and could thus be hypothesized to have *the same*, albeit unknown, label. Domain adaptation in a purely unsupervised setting (no labeled target domain examples) has been considered by Gopalan et al. (2011) and Gong et al. (2012). The main idea of both works is to build subspaces for the source as well as the target domain and to consider the path between them on the corresponding manifold. A new feature representation is calculated by concatenating intermediate subspaces on the path. Whereas, Gopalan et al. (2011) samples a finite set of intermediate subspaces, the Geodesic Flow Kernel (**gfk**) of Gong et al. (2012) shows how to use all subspaces on the geodesic path by using a kernel trick. This yields a symmetric kernel for source and target points that can be used for example for nearest neighbor classification. More recently, Chopra et al. (2013) extended this framework to handle image features learned using deep convolutional neural networks.

3 Category Invariant Feature Transformations

Our first approach is to learn a single transformation matrix $\boldsymbol{W}$ which maps examples between the source and target domains. The objective for the transformation is to diminish

domain-induced differences so that examples can be compared directly. This mapping step can then be followed by standard distance-based learning.

We denote the source domain as $\mathcal{X}$ and the target domain as $\mathcal{Z}$. Similarly, we denote the source data points as $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_{n_\mathcal{X}}] \in \mathbb{R}^{d_\mathcal{X} \times n_\mathcal{X}}$ with labels $\boldsymbol{y} = [y_1, \ldots, y_{n_\mathcal{X}}]$ and the target data points as $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_{n_\mathcal{Z}}] \in \mathbb{R}^{d_\mathcal{Z} \times n_\mathcal{Z}}$ with corresponding labels $\boldsymbol{h} = [h_1, \ldots, h_{n_\mathcal{Z}}]$. The target domain is assumed to have significantly fewer labeled examples than the source, i.e. $n_\mathcal{Z} \ll n_\mathcal{X}$, and there may even be some categories for which the target domain has *no* labeled examples. We use whichever categories have labeled target examples to learn a transformation that is generalizable across categories and so can be applied to *all* categories at test time. Note that our algorithm allows the source and target feature spaces to have different dimensions ($d_\mathcal{X} \neq d_\mathcal{Z}$).

To learn such a transformation we will define a matrix regularizer, $r(\boldsymbol{W})$ and a loss, $\mathcal{L}(\boldsymbol{W}, \mathbf{X}, \mathbf{Z}, \boldsymbol{y}, \boldsymbol{h})$, which are computed as some function of the category labeled source and target data.[1] With these two terms defined we solve the following general optimization problem:

$$\hat{\boldsymbol{W}} = \arg\min_{\boldsymbol{W}} \; r(\boldsymbol{W}) + \lambda \cdot \mathcal{L}(\boldsymbol{W}, \mathbf{X}, \mathbf{Z}, \boldsymbol{y}, \boldsymbol{h}) \tag{1}$$

We present multiple learning algorithms that arise from using different regularizers and loss functions within the framework of Eq. (1).

4 Category Invariant Feature Transformations Through Similarity Constraints

Learning a transformation can be also viewed as learning a similarity function between source and target points, $\text{sim}(\boldsymbol{W}, \mathbf{x}, \mathbf{z}) = \mathbf{x}^T \boldsymbol{W} \mathbf{z}$. This perspective allows us to use metric learning techniques (Davis et al. 2007) and to extend them towards a domain adaptation scenario. Intuitively, a desirable property of this similarity function is that it should have a high value when the source and target points are of the same category and a low value when the source and target points are of different categories.

This intuitive goal can be formulated by constructing a constraint for each pair $(\{\mathbf{x}, y\}, \{\mathbf{z}, h\})$ of labeled source and target points:

$$c(\boldsymbol{W}, \mathbf{x}, \mathbf{z}, y, h) := \begin{cases} \text{sim}(\boldsymbol{W}, \mathbf{x}, \mathbf{z}) > u & y = h \\ \text{sim}(\boldsymbol{W}, \mathbf{x}, \mathbf{z}) < l & y \neq h \end{cases}, \tag{2}$$

for some constants, $u, l \in \mathbb{R}$.

[1] Note that in general we could equally optimize a second loss function between the source and target data which considers *instance* level constraints. However, to distinguish ourselves from prior work which focused on learning a metric requiring instance constraints, we present our algorithms assuming only category level information to demonstrate the effectiveness of using only this coarser level of supervision.

If optimized, the constraints specified in Eq. (2) guarantee that source and target points with the same label have high similarity and that source and target points with different labels have low similarity.

In general, we do not need each pairwise constraint to be satisfied to learn a good similarity function, therefore, we optimize soft constraints in the form of the following loss function:

$$\ell(\boldsymbol{W}, \mathbf{x}, \mathbf{z}, y, h) = \begin{cases} \max(0, \text{sim}(\boldsymbol{W}, \mathbf{x}, \mathbf{z}) - u) \\ \qquad \text{if } y = h \\ \max(0, l - \text{sim}(\boldsymbol{W}, \mathbf{x}, \mathbf{z})) \\ \qquad \text{if } y \neq h \end{cases} \tag{3}$$

Finally, we define a loss for all labeled data points as the squared sum over each pairwise loss:

$$\mathcal{L}(\boldsymbol{W}, \mathbf{X}, \mathbf{Z}, \boldsymbol{y}, \boldsymbol{h}) = \sum_{i,j} [\ell(\boldsymbol{W}, \mathbf{x}_i, \mathbf{z}_j, y_i, h_j)]^2 \; . \tag{4}$$

Using this loss function in the general framework of Eq. (1), we seek to solve the following optimization problem:

$$\hat{\boldsymbol{W}} = \arg\min_{\boldsymbol{W}} \; r(\boldsymbol{W}) + \sum_{i,j} [\ell(\boldsymbol{W}, \mathbf{x}_i, \mathbf{z}_j, y_i, h_j)]^2 \; . \tag{5}$$

Constraints thus far have been defined for category level correspondences, however, if additional paired instance correspondence is available for some data, this auxiliary information could be incorporated using the same similarity constraint technique. In Donahue et al. (2013), we also showed that constraints between labeled and unlabeled target points can help learning in a semi-supervised fashion.

An important second part of the objective function as defined in Sect. 3 is the regularizer of the transformation matrix. This term contains our prior knowledge about the transformation and has to be chosen carefully. We present two types of very flexible regularization terms in the following sections.

4.1 LogDet Regularizer for Symmetric Transforms

We will begin by considering the log determinant (LogDet) regularizer:

$$r(\boldsymbol{W}) = \text{tr}(\boldsymbol{W}) - \log\det(\boldsymbol{W}) \tag{6}$$

for positive definite matrices $\boldsymbol{W}$.

Using the LogDet regularizer causes the formulation in Eq. (5) to become equivalent to that of Information-theoretic Metric Learning (ITML), which indirectly learns a transformation matrix $\boldsymbol{W}$ corresponding to a linear transformation between $\mathcal{X}$ and $\mathcal{Z}$ by optimizing the pairwise loss functions given in Eq. (5).

With this regularization term, the optimization function is kernelizable and a final non-linear transformation can be learned to map between the source and target points.

While this model is intuitively appealing for domain adaptation, it requires a key simplifying assumption that the source and target data have the same dimension; i.e., $d_{\mathcal{X}} = d_{\mathcal{Z}}$. This follows from the fact that the matrix trace and determinant are only defined for square matrices $\boldsymbol{W}$. An even stronger restriction on $\boldsymbol{W}$ made by the LogDet regularizer is that it is only defined over symmetric positive definite matrices. Implicitly, if $\boldsymbol{W}$ is positive definite then $\boldsymbol{W}$ can be decomposed into the product of two identical matrices - $\boldsymbol{W} = \boldsymbol{R}^T\boldsymbol{R}$. Therefore, the similarity function learned can be decomposed into:

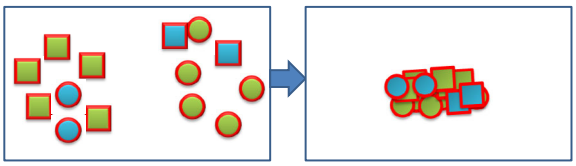
$$\text{sim}(\boldsymbol{W}, \mathbf{x}, \mathbf{z}) = \mathbf{x}^T\boldsymbol{W}\mathbf{z} = (\boldsymbol{R}\mathbf{x})^T(\boldsymbol{R}\mathbf{z}) \tag{7}$$

The observation here is that a symmetric positive definite matrix corresponds to applying the same transformation to the source and target domains.

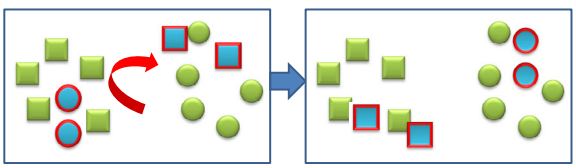
Using the LogDet regularizer limits the applicability of domain adaptation due to the restricted class of possible transformation matrices $\boldsymbol{W}$. Consider the scenario in Fig. 2, where there is no symmetric transformation that can transform between the source and target domains. In the next section, we will mitigate this limitation.

4.2 Frobenius Regularizer for Asymmetric Transforms

In order to avoid the restrictions of the symmetric transformation model for adaptation, we seek an alternative regularizer that allows the model to be applied to domains of differing dimensionalities but that still retains the benefits of kernelization. We choose the Frobenius norm regularizer, which is defined for general matrices $\boldsymbol{W}$ in asymmetric transformations (Fig. 2). We call this problem the **Asymmetric Regularized Cross-domain transformation** problem with similarity and dissimilarity constraints, or **arc-t** for short, in the rest of the paper.



(a) A symmetric transformation – the same rotation and scaling applied to both domains (green and blue) – cannot separate classes (circles and squares)



(b) An asymmetric transformation – a rotation applied only to blue domain – successfully compensates for domain shift

Fig. 2 A conceptual illustration of how an asymmetric domain transformation matrix corresponding to a linear transformation can be more flexible than a symmetric one (Color figure online)

Using the loss function defined in Eq. (5), our new optimization objective is given as follows:

$$\min_{\boldsymbol{W}} \ \frac{1}{2}\|\boldsymbol{W}\|_F^2 + \lambda \sum_{i,j}[\ell(\boldsymbol{W}, \mathbf{x}_i, \mathbf{z}_j, y_i, h_j)]^2 \tag{8}$$

There are two main limitations of the transformation learning problem (8) presented above. First, it is limited to linear transformation matrices, which may not be sufficient for some adaptation tasks. Second, the size of $\boldsymbol{W}$ grows as $d_{\mathcal{X}} \cdot d_{\mathcal{Z}}$, which may be prohibitively large for some problems. In the next section, we present a kernelization result that overcomes both of these shortcomings.

4.3 Kernelization Analysis

In this section, we prove that (5) may be solved in kernel space *for a wide class of regularizers*,[2] resulting in non-linear transformation matrices whose complexity is independent of the dimensions of the points in either domain. This kernelization result is critical to obtaining good performance for several domain adaptation tasks. Note that kernelization has been proven for some metric learning formulations, such as Kulis et al. (2009); in all these cases, the kernelization results assume that $\boldsymbol{W}$ is symmetric positive definite, whereas our results hold for arbitrary $\boldsymbol{W}$. We also note connections to the work of Argyriou et al. (2010), which derives representer theorems for various matrix learning problems. However, they do not consider domain adaptation, and are mainly concerned with theoretical results for matrix learning problems such as collaborative filtering and multi-task learning.

The main idea behind the following result is to show that (1) the learned similarity function resulting from solving Eq. (5) can be computed only using inner products between data points in the source domain and inner products between data points in the target domain, and (2) Eq. (5) can be reformulated as an optimization problem involving such inner products and whose size is independent of the dimensions $d_{\mathcal{X}}$ and $d_{\mathcal{Z}}$. Then we can replace standard inner products with arbitrary kernel functions such as the Gaussian RBF

[2] Note that we present this result for the specific case of using the Frobenius norm regularizer, though in fact our analysis holds for the class of regularizers $r(\boldsymbol{W})$ that can be written in terms of the singular values of $\boldsymbol{W}$; that is, if $\sigma_1, \ldots, \sigma_p$ are the singular values of $\boldsymbol{W}$, then $r(\boldsymbol{W})$ is of the form $\sum_{j=1}^{p} r_j(\sigma_j)$ for some scalar functions r_j, which is globally minimized by zero. For example, the squared Frobenius norm $r(\boldsymbol{W}) = \frac{1}{2}\|\boldsymbol{W}\|_F^2$ is a special case where $r_j(\sigma_j) = \frac{1}{2}\sigma_j^2$.

kernel function, resulting in non-linear learned transformations between the input space of the source domain and the input space of the target domain. In the following analysis, the input kernel matrices over within-domain points are given as $K_{\mathcal{X}} = \mathbf{X}^T\mathbf{X}$ and $K_{\mathcal{Z}} = \mathbf{Z}^T\mathbf{Z}$. We begin with the first result (proof in "Appendix").

Lemma 1 *Assume that $K_{\mathcal{X}}$ and $K_{\mathcal{Z}}$ are strictly positive definite.*[3]

Then there exists an $n_{\mathcal{X}} \times n_{\mathcal{Z}}$ matrix $\boldsymbol{L}$ such that the optimal solution $\boldsymbol{W}^*$ to (5) is of the form $\boldsymbol{W}^* = \mathbf{X}\boldsymbol{K}_{\mathcal{X}}^{-1/2}\boldsymbol{L}\boldsymbol{K}_{\mathcal{Z}}^{-1/2}\mathbf{Z}^T$.

The above result demonstrates the existence of such a matrix $\boldsymbol{L}$, which is important to reformulate the optimization problem in eq. (8) in terms of $\boldsymbol{L}$ in the following. Furthermore, one important consequence of the above lemma is that, given arbitrary points $\mathbf{x}$ and $\mathbf{z}$, the function $\text{sim}(\boldsymbol{W}, \mathbf{x}, \mathbf{z})$ can be computed in kernel space—by replacing $\boldsymbol{W}$ with $\mathbf{X}\boldsymbol{K}_{\mathcal{X}}^{-1/2}\boldsymbol{L}\boldsymbol{K}_{\mathcal{Z}}^{-1/2}\mathbf{Z}^T$, the expression $\mathbf{x}^T\boldsymbol{W}\mathbf{z}$ can be written purely in terms of inner products.

Additionally, if $\boldsymbol{W}$ was a square matrix, then an extended analysis would also hold for regularizations of the form $||\boldsymbol{W} - \boldsymbol{A}||_F^2$, where $\boldsymbol{A}$ is some known matrix, for example the identity matrix. Regularizations of this form may be useful if there exists some prior knowledge about the domain shift or if $\boldsymbol{W}$ is assumed to be close to the identity matrix, in which case $\boldsymbol{A}$ is set to $\boldsymbol{I}$.

Using the above lemma, we now show how to equivalently rewrite the optimization (8) in terms of the kernel matrices $\boldsymbol{K}_{\mathcal{X}}$ and $\boldsymbol{K}_{\mathcal{Z}}$ to solve for $\boldsymbol{L}$ (proof in "Appendix"):

Theorem 1 *Assume the conditions of Lemma 1 hold. If $\boldsymbol{W}^*$ is the optimal solution to (8) and $\boldsymbol{L}^*$ is the optimal solution to the following problem:*

$$\min_{\boldsymbol{L}} r(\boldsymbol{L}) + \mathcal{L}\left(\boldsymbol{L}, \boldsymbol{K}_{\mathcal{X}}^{1/2}, \boldsymbol{K}_{\mathcal{Z}}^{1/2}, \boldsymbol{y}, \boldsymbol{h}\right) \tag{9}$$

then $\boldsymbol{W}^* = \mathbf{X}\boldsymbol{K}_{\mathcal{X}}^{-1/2}\boldsymbol{L}^*\boldsymbol{K}_{\mathcal{Z}}^{-1/2}\mathbf{Z}^T$.

To summarize, Theorem 1 demonstrates that, instead of solving (8) for $\boldsymbol{W}$ directly, we can *equivalently* solve (9) for $\boldsymbol{L}$, and then implicitly construct $\boldsymbol{W}$ via $\boldsymbol{W} = \mathbf{X}\boldsymbol{K}_{\mathcal{X}}^{-1/2}\boldsymbol{L}\boldsymbol{K}_{\mathcal{Z}}^{-1/2}\mathbf{Z}^T$. In particular, this form of $\boldsymbol{W}$ allows us to compute $\mathbf{x}^T\boldsymbol{W}\mathbf{z}$ using only kernel functions. Though our analysis focuses on one particular regularizer—the squared Frobenius norm—one can imagine applying our analysis to other regularizers. For example, the trace norm $r(\boldsymbol{W}) = \text{tr}(\boldsymbol{W})$ also falls under our framework; because the trace norm as a regularizer is known to produce low-rank matrices $\boldsymbol{W}$, it would be desirable in kernel dimension-reduction settings. In showing kernelization for this regularizer, we actually prove a much stronger result, namely that kernelization holds for a large class of regularizers that includes the squared Frobenius norm and other regularizers, as discussed in Sect. 4.3.

Whether using the linear or kernelized version of the algorithm, the general idea of using pairwise constraints to learn $\boldsymbol{W}$ limits the ability of this learning algorithm to scale with the number of labeled points in the source and target, since the number of constraints generated is $n_{\mathcal{X}} \cdot n_{\mathcal{Z}}$. Additionally, $\boldsymbol{W}$ is learned so as to place source and target points close if they are of the same category and far if they are from different categories. While this is an intuitive notion, it fails to directly optimize the overall objective of correctly classifying target points.

In the next section, we describe an alternate approach which overcomes these limitation by jointly learning $\boldsymbol{W}$ and classifier parameters.

4.4 Optimization

In this section, we briefly describe optimization techniques that can be used to solve the objectives described.

When optimizing the objective with a LogDet norm regularizer, we use the standard ITML method as mentioned in Sect. 4.1. To optimize the objective with the Frobenius norm regularizer, we use one of two methods. First, note that the problem can actually be reformulated as a quadratic programming (QP) problem, after which any standard QP solver can be used to optimize our objective. However, if the size of $\boldsymbol{W}$ is too large, this is impractical. A separate approach is to use the Bregman divergence method (Davis et al. 2007). Both techniques yield similar performance, but have varying convergence rates.

5 Category Invariant Feature Transformations Through Optimizing Classification Objective

In this section, we present a different loss function that can be used within the transform-based domain adaptation framework defined in Eq. (1). The goal now is to directly optimize a classification objective for the target points, while simultaneously presenting a learning algorithm that is more scalable with the number of labeled source and target points (Fig. 3).

For our algorithm, we consider linear hyperplane classifiers. For example, assume that a one-versus-all linear SVM classifier has been trained on the labeled source data over all K categories. Let $\boldsymbol{\theta}_k$ denote the normal to the hyperplane associated with the k'th binary SVM problem. Similarly, let b_k be the offset to the hyperplane associated with the k'th

[3] The assumption that the kernel matrices are strictly positive definite is not a severe limitation. For the Gaussian RBF kernel, strict positive definiteness can always be assured and for other kernel functions, the matrices can be regularized by adding a scaled identity matrix.

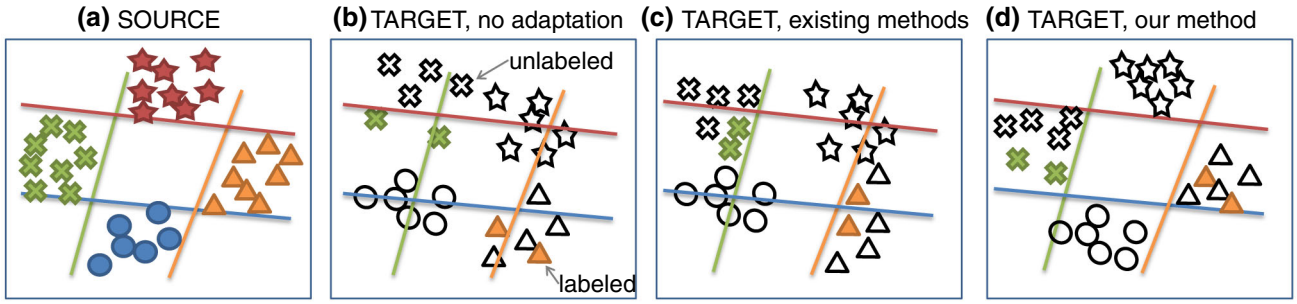


Fig. 3 (**a**) Linear classifiers (shown as decision boundaries) learned for a four-class problem on a fully labeled source domain. (**b**) Problem: classifiers learned on the source domain do not fit the target domain points shown here due to a change in feature distribution. (**c**) Existing SVM-based methods only adapt the features of classes with labels (crosses and triangles). (**d**) Our method adapts all points, including those from classes without labels, by transforming all target features to a new domain-invariant representation

binary SVM problem. Finally, let $\tilde{\boldsymbol{\theta}}_k^T = \left[\boldsymbol{\theta}_k^T \; b_k\right]$ be the full affine hyperplane representation.

Intuitively, we seek to learn a transformation matrix $\boldsymbol{W}$ such that once $\boldsymbol{W}$ is applied to the target points, they will be classified accurately by the source SVM. We consider learning an affine linear transformation matrix, which can be easily done using homogeneous coordinates for our data points: $\tilde{\mathbf{z}}^T = \left[\mathbf{z}^T \; 1\right]$.

The key idea is now to use constraints based on linear classifiers instead of single instances. In particular, we require that transformed target points are correctly classified in the source domain:

$$c(\boldsymbol{W}, \tilde{\boldsymbol{\theta}}_k, \tilde{\mathbf{z}}, h) := \mathbb{I}(h = k)\left(\tilde{\boldsymbol{\theta}}_k^T \boldsymbol{W} \tilde{\mathbf{z}}\right) \geq 1 \;, \tag{10}$$

where $\mathbb{I}$ is the signed indicator function, with $\mathbb{I}(z) = 1$ when z is true and $\mathbb{I}(z) = -1$ in the other case. If Eq. (10) is fulfilled, all transformed target points would be correctly classified by the source linear classifier. However, this is only possible for separable cases, so instead we optimize the soft constraints in form of the hinge loss:

$$\ell(\boldsymbol{W}, \tilde{\boldsymbol{\theta}}_k, \mathbf{z}_i, h_i) = \max\left(0, 1 - \mathbb{I}(h_i = k) \cdot \tilde{\boldsymbol{\theta}}_k^T \boldsymbol{W} \tilde{\mathbf{z}}_i\right) \tag{11}$$

Similarly, if we use $\boldsymbol{\Theta} = \left[\tilde{\boldsymbol{\theta}}_1 \ldots \tilde{\boldsymbol{\theta}}_K\right]$ to denote all hyperplane parameters of the one-versus-all classifier, the loss over all target points and all categories is given as:

$$\mathcal{L}(\boldsymbol{W}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{h}) = \sum_{k,i} \ell(\boldsymbol{W}, \tilde{\boldsymbol{\theta}}_k, \mathbf{z}_i, h_i) \tag{12}$$

Because the target points are transformed into the source domain space with $\boldsymbol{W}$, we simply define the source data term in our loss function as standard SVM hinge loss summed over all categories:

$$\mathcal{L}(\boldsymbol{\Theta}, \mathbf{X}, \boldsymbol{y}) = \sum_{k,i} \max\left(0, 1 - \mathbb{I}(y_i = k) \cdot \tilde{\boldsymbol{\theta}}_k^T \mathbf{x}_i\right) \tag{13}$$

Once the transformation matrix $\boldsymbol{W}$ has been learned, we can also use it to transform linear classifiers $\tilde{\boldsymbol{\theta}}_k$ to the target domain that had been learned with source data only. This is a huge advantage of modelling the domain shift as being category-invariant, because we only need a few categories present in both target and source training data and are able to transfer all available category models in the source domain to the target domain. For regularization of $\boldsymbol{W}$, we use the Frobenius norm regularizer for this optimization problem. Optimizing this objective in Eq. (1) using the loss in Eq. (11) and the Frobenius norm regularizer leads to a category invariant and asymmetric transformation matrix, which considers classifier constraints in the source domain. Additionally, the learning algorithm no longer has a linear dependency on the number of source training examples and instead scales with the number of categories and the number of labeled target points, $K \cdot n_{\mathcal{Z}}$.

6 Jointly Optimizing Classifier and Transformation

Our goal in this section is to jointly learn (1) affine hyperplanes that separate the categories in the common domain consisting of the source domain and target points projected to the source and (2) the new feature representation of the target domain determined by the transformation matrix $\boldsymbol{W}$ mapping points from the target domain into the source domain.

The algorithm and the change of constraints presented in the previous section is especially useful when linear classifiers are already learned in the source domain. However, we can also formulate a joint learning problem for the transformation matrix and the classifier parameters; i.e., the hyperplane parameters and thus the decision boundary are also affected by the additional training data provided from the target domain.

The transformation matrix should have the property that it projects the target points onto the correct side of each

source hyperplane and the joint optimization also maximizes the margin between two classes. Therefore, we refer to this method as **Maximum Margin Domain Transform**, or **mmdt**.

The joint optimization problem can be formulated by adding a regularizer on $\boldsymbol{\Theta}$.

$$\min_{\boldsymbol{W},\boldsymbol{\Theta}} \frac{1}{2}\|\boldsymbol{W}\|_F^2 + \frac{1}{2}\|\boldsymbol{\Theta}\|_F^2 + \lambda\,\mathcal{L}(\boldsymbol{W},\boldsymbol{\Theta},\mathbf{Z},\boldsymbol{h}) + \lambda_{\mathcal{X}}\,\mathcal{L}(\boldsymbol{\Theta},\mathbf{X},\boldsymbol{y}) \tag{14}$$

In contrast to the previous optimization problems, the problem in Eq. (14) is no longer convex. For this reason, we perform coordinate gradient descent by alternating between optimizing with respect to $\boldsymbol{W}$ and $\boldsymbol{\Theta}$:

1. Initialize $\boldsymbol{\Theta}^0$ using a 1-vs-all SVM trained on the source data only.
2. Learn $\boldsymbol{W}^t$ assuming fixed $\boldsymbol{\Theta}^t$.
3. Learn $\boldsymbol{\Theta}^{t+1}$ assuming fixed $\boldsymbol{W}^t$.
4. Iterate between (2)-(3), until convergence.

Note that step (2) is equivalent to solving the optimization problem presented in Sect. 5. Additionally, note that step (3) is equivalent to solving a multi-category SVM problem defined over source and transformed target data points. This can again be solved using K 1-vs-all binary SVM classifiers.

An important property of the alternating optimization is that we can indeed prove convergence by exploiting the convexity of both sub-problems.

Lemma 2 *Steps (2) and (3) will never increase the complete joint objective function.*

Proof Let $J(\boldsymbol{W},\boldsymbol{\Theta})$ denote the value of the joint objective function.

Claim 1 $J\left(\boldsymbol{W}^t,\boldsymbol{\Theta}^t\right) \geq J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}^t\right)$

$$J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}^t\right) = \min_{\boldsymbol{W}} J\left(\boldsymbol{W},\boldsymbol{\Theta}^t\right) \leq J\left(\boldsymbol{W}^t,\boldsymbol{\Theta}^t\right)$$

Claim 2 $J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}^t\right) \geq J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}^{t+1}\right)$

$$J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}^{t+1}\right) = \min_{\boldsymbol{\Theta}} J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}\right) \leq J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}^t\right)$$

The key here is that steps (2) and (3) of our algorithm are convex optimization problems and so we know that each objective will never increase as the new variable values are learned.

Theorem 2 *The joint objective function for Eq.* (14) *will converge.*

Proof Using Lemma (2), we can directly show that the joint objective function will not increase from one iteration to the next:

$$J(\boldsymbol{W}^t,\boldsymbol{\Theta}^t) \geq J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}^t\right) \geq J\left(\boldsymbol{W}^{t+1},\boldsymbol{\Theta}^{t+1}\right)$$

Additionally, since the joint objective is lower bounded by zero, this proves that the joint objective will converge for a sufficiently small step size if optimizing using gradient descent. □

It is important to note that since both steps of our iterative algorithm can be solved using standard QP solvers, the algorithm can be easily implemented. Furthermore, we also developed a fast optimization technique based on dual coordinate descent and exploiting an implicit rank contraint of $\boldsymbol{W}$ in Rodner et al. (2013). The method allows using the MMDT algorithm even in large-scale scenarios with tens of thousands of examples and high-dimensional features, because not all of the entries of $\boldsymbol{W}$ have to be optimized.

7 Analysis

We now analyze and compare the proposed algorithms against each other and the previous feature transform methods **hfa** (Duan et al. 2012a) and **gfk** (Gong et al. 2012). Comparisons are summarized in Table 1.

The **arc-t** formulation, of Sect. 4, has two distinct limitations. First, it must solve $n_{\mathcal{X}} \cdot n_{\mathcal{Z}}$ constraints, whereas **mmdt**, of Sect. 6, only needs to solve $K \cdot n_{\mathcal{Z}}$ constraints, for a K category problem. In general, **mmdt** scales to much larger source domains than **arc-t.** The second benefit of the **mmdt** learning approach is that the transformation matrix learned using the max-margin constraints is learned jointly with the classifier, and explicitly seeks to optimize the final SVM classifier objective. While **arc-t**'s similarity-based constraints seek to map points of the same category arbitrarily close to one another, followed by a separate classifier learning step, **mmdt** seeks simply to project the target points onto the correct side of the learned hyperplane, leading to better classification performance.

The **hfa** formulation (Duan et al. 2012a) also takes advantage of the max-margin framework to directly optimize the classification objective while learning transformation matrices. **hfa** learns the classifier and transformations to a common latent feature representation between the source and target. However, **hfa** is formulated to solve a binary problem so a new feature transformation must be learned for each category. Therefore, unlike **mmdt**, **hfa** cannot learn a representation that generalizes to novel target categories. Additionally, due to the difficulty of defining the dimension of the latent feature representation directly, the authors optimize with respect to a larger combined transformation matrix and a relaxed constraint. This transformation matrix becomes too large when

Table 1 Unlike previous methods (**hfa** by Duan et al. 2012a and **gfk** by Gong et al. 2012), our final approach using max-margin constraints and frobenius norm regularizer is able to simultaneously learn muti-category representations that can transfer to novel classes, scale to large training datasets, and handle different feature dimensionalities

	hfa	gfk	**symm**	**arc-t**	**mmdt**
Multi-class	No	**Yes**	**Yes**	**Yes**	**Yes**
Large datasets	No	**Yes**	No	No	**Yes**
Heterogeneous features	**Yes**	No	No	**Yes**	**Yes**
Optimize max-margin objective	**Yes**	No	no	no	**Yes**

the feature dimensions in source and target are large, so the **hfa** problem must usually be solved in kernel space. This can make the method slow and cause it to scale poorly with the number of training examples. In contrast, **mmdt** can be efficiently solved in linear feature space which makes it fast and potentially more scalable.

Finally, **gfk** (Gong et al. 2012) formulates a kernelized representation of the data that is equivalent to computing the dot product in infinitely many subspaces along the geodesic flow between the source and target domain subspaces. The kernel is defined to be symmetric, so it cannot handle source and target domains of different initial dimension. Additionally, **gfk** does not directly optimize a classification objective. In contrast, **mmdt** can handle source and target domains of different feature dimensions via an asymmetric $\boldsymbol{W}$, as well as directly optimizing the classification objective.

8 Domain Adaptation Datasets

We begin by introducing the data on which we will evaluate our algorithms.

8.1 *Office* Database

In most of our experiments, we consider the *Office* database first introduced by Saenko et al. (2010), which has become the de facto standard for benchmarking visual domain adaptation methods. This database allows researchers to study, evaluate and compare solutions to the domain shift problem by establishing a multiple-domain labeled dataset and benchmark. In addition to the domain shift aspects, this database also proposes a challenging office environment category learning task which reflects the difficulty of real-world indoor robotic object recognition. It contains images originating from the following three domains:

Images from the web: The first domain, *amazon*, consists of images downloaded from online merchants (www.amazon.com). These images are of products shot at medium resolution typically taken in an environment with studio lighting conditions. The *amazon* domain contains 31 categories with an average of 90 images each. The images capture the large intra-class variation of these categories, but typically show the objects only from a canonical viewpoint.

Images from a digital SLR camera: The second domain, *dslr*, consists of images that are captured with a digital SLR camera in realistic environments with natural lighting conditions. The images have high resolution ($4{,}288 \times 2{,}848$) and low noise. *dslr* has images of the 31 object categories, with five different objects for each, in an office environment. Each object was captured with on average three images taken from different viewpoints, for a total of 423 images.

Images from a webcam: The third domain, *webcam*, consists of images of the 31 categories recorded with a simple webcam. The images are of low resolution (640×480) and show significant noise and color as well as white balance artifacts. Many current imagers on robotic platforms share a similarly-sized sensor, and therefore also possess these sensing characteristics. The resulting *webcam* dataset contains the same 5 objects per category as in *dslr*, for a total of 795 images.

The database represents several interesting visual domain shifts. It allows us to investigate the adaptation of category models learned on the web to SLR and webcam images, which can be thought of as in situ observations on a robotic platform in a realistic office or home environment. Furthermore, domain transfer between the high-quality DSLR images to low-resolution webcam images allows for a very controlled investigation of category model adaptation, as the same objects were recorded in both domains.

The *Office* dataset images are available together with SURF BoW features that are vector quantized to 800 dimensions. We use these features in all experiments except where explicitly indicated otherwise.

We also use a version of the *Office* dataset, available from Gong et al. (2012), which consists of the 10 categories from the *Office* dataset that also appear in *Caltech256*. The same SURF BoW 800-dimensional features are available for the *Caltech256* images.

8.2 Large-scale Database

We also demonstrate the efficiency of our domain adaptation methods in a large-scale setting (Sect. 9.4). For this purpose,

we consider two domains. The source domain, the *Bing* dataset (Bergamo and Torresani 2010), consists of images obtained using the Bing search engine. In our experiments, we train on 50 source domain examples per category. The target domain is a subset of the images in the *Caltech-256* benchmark dataset. We vary the number of target domain examples from 5 to 20.

Note that we use the original features (Classeme 2625 dimensional) and train/test splits introduced by Bergamo and Torresani (2010).

9 Experiments

In the following, we evaluate our methods on the datasets described in the previous section and compare the results to state-of-the-art supervised domain adaptation methods in different domain adaptation scenarios. In particular, we compare against the following methods in the experiments where applicable:

svm$_s$ A support vector machine using source training data.
svm$_t$ A support vector machine using target training data.
hfa A max-margin transform approach that learns a latent common space between source and target as well as a classifier that can be applied to points in that common space (Duan et al. 2012a).
gfk The geodesic flow kernel proposed by Gong et al. (2012) applied to all source and target data (including test data). Following Gong et al. (2012), we use a 1-nearest neighbor classifier with the geodesic flow kernel.

9.1 Standard Supervised Domain Adaptation

In our first set of experiments, we use the 10 category subset of the *Office* database, together with the same 10 categories available from the *Caltech* dataset, to evaluate multi-class accuracy in the standard domain adaptation setting where a few labeled examples are available for all categories in the target domain. We follow the setup of Saenko et al. (2010) and Gong et al. (2012): 20 training examples for *amazon* source (8 for other source domains) and 3 labeled examples per category for the target domain. We created 20 random train/test splits and averaged the results across them.

The multi-class accuracy for each domain pair is shown in Table 2. Our **mmdt** method is the top performing overall, achieving 52.5 % accuracy averaged over the 12 domain shifts we explored. This result may be somewhat surprising, because **mmdt** encodes no knowledge of the feature representation, but on shifts where features are homogeneous, still outperforms methods like **gfk** and **symm** which *assume* feature homogeneity. This demonstrates the strength of **mmdt** as a generic domain adaptation approach.

Looking at individual domain shifts, we see that **mmdt** outperforms all other methods in 6 out of the 12 domain shifts. Of the results on the *Office* dataset only (the first 6 rows of Table 2), **mmdt** performs the best when either the source or target domain is *amazon*. Because the shift between *amazon* and either of the other two *Office* domains (*dslr* and

Table 2 Multi-class accuracy for the standard supervised domain adaptation setting

	Baselines				Our methods		
	svm$_s$	**svm**$_t$	**hfa**	**gfk**	**symm**	**arc-t**	**mmdt**
a → w	33.9 ± 0.7	62.4 ± 0.9	61.8 ± 1.1	58.6 ± 1.0	51.0 ± 1.4	55.7 ± 0.9	**64.6 ± 1.2**
a → d	35.0 ± 0.8	*55.9 ± 0.8*	52.7 ± 0.9	50.7 ± 0.8	47.9± 1.4	50.2 ± 0.7	**56.7 ± 1.3**
w → a	35.7 ± 0.4	45.6 ± 0.7	45.9 ± 0.7	44.1 ± 0.4	43.7 ± 0.7	43.4 ± 0.5	**47.7 ± 0.9**
w → d	66.6 ± 0.7	55.1 ± 0.8	51.7 ± 1.0	*70.5 ± 0.7*	69.8± 1.0	**71.3 ± 0.8**	67.0 ±1.1
d → a	34.0 ± 0.3	*45.7 ± 0.9*	*45.8 ± 0.9*	*45.7 ± 0.6*	42.7 ± 0.5	42.5 ± 0.5	**46.9 ± 1.0**
d → w	74.3 ± 0.5	62.1 ± 0.8	62.1 ± 0.7	76.5 ± 0.5	**78.4 ± 0.9**	**78.3 ± 0.5**	74.1 ±0.8
a → c	35.1 ± 0.3	32.0 ± 0.8	31.1 ± 0.6	36.0 ± 0.5	**39.1 ± 0.5**	37.0 ± 0.4	36.4 ± 0.8
w → c	31.3 ± 0.4	30.4 ± 0.7	29.4 ± 0.6	31.1 ± 0.6	**34.0 ± 0.5**	31.9 ± 0.5	32.2 ± 0.8
d → c	31.4 ± 0.3	31.7 ± 0.6	31.0 ± 0.5	32.9 ± 0.5	**34.9 ± 0.4**	*33.5 ± 0.4*	*34.1 ± 0.8*
c → a	35.9 ± 0.4	45.3 ± 0.9	45.5 ± 0.9	44.7 ± 0.8	43.8 ± 0.6	44.1 ± 0.6	**49.4 ± 0.8**
c → w	30.8 ± 1.1	60.3 ± 1.0	60.5 ± 0.9	**63.7 ± 0.8**	50.5± 1.6	55.9 ± 1.0	**63.8 ± 1.1**
c → d	35.6 ± 0.7	55.8 ± 0.9	51.9 ± 1.1	**57.7 ± 1.1**	48.6± 1.1	50.6 ± 0.8	*56.5 ± 0.9*
mean	40.0 ± 0.6	48.5 ± 0.8	47.4 ± 0.8	*51.0 ± 0.7*	48.7 ± 0.9	49.5 ± 0.6	**52.5 ± 1.0**

All results are from our implementation. When averaged across all domain shifts the reported average value for **gfk** was 51.65 while our implementation had an average of 51.0 ± 0.7. Therefore, the result difference is well within the standard deviation over data splits. Bold indicates the best result for each domain split. Italic indicates the group of results that are close to the best-performing result. The domain names are shortened for space: a: *amazon*, w: *webcam*, d: *dslr*, c: *caltech*

webcam) is much more significant than the shift between *dslr* and *webcam*, as indicated by the large performance discrepancy between *amazon* and non-*amazon* shifts with the $\mathbf{svm}_S$ method, this result indicates that **mmdt** is particularly well-suited to handling larger domain shifts.

Our other methods, **symm** and **arc-t**, have better performance than **mmdt** (and all baselines) on the *webcam* and *dslr* shifts. This demonstrates the utility of these methods in learning smaller domain shifts. Their higher relative performance on such tasks might be due to their cross-domain pairwise constraints on individual examples, which may be less meaningful in cases when the domain shift is larger and individual pairs of examples from a particular category are unlikely to correspond. The **gfk** baseline also performs well on the *webcam* and *dslr* shifts. This fits with our intuition since **gfk** is a 1-nearest neighbor approach and, as such, is more suitable when the domains are initially similar.

In the *caltech* results (the last six rows of Table 2), we see that the task overall is much easier when *caltech* is the source domain than when it is the target domain, indicating that the *caltech* data is more valuable for recognition in the *Office* domains than the *Office* data is for recognition of the *caltech* categories. When *caltech* is the target domain, the more difficult of the two situations, our **symm** method outperforms all others. On the other hand, when *caltech* is the source domain, we see the best performance from our **mmdt** method and the **gfk** baseline, with our **arc-t** method performing somewhere in between in most cases. This seems to indicate that **mmdt** is the best of the methods explored when working with a very rich source domain (at least relative to the target domains) like *caltech*, whereas **symm** is superior when the source domain is more homogeneous like the *Office* domains.

For completeness, we additionally experimented with a multi-view baseline, Generalized Multiview Analysis (GMA) (Sharma et al. 2012). To apply this method here we computed the mean for each category in the source and target datasets. Then, for each category k, we considered the mean feature vector from the source and the mean feature vector from the target to be multiple views of the category, k. Using this technique we found that GMA had a mean classification performance across all shifts of 45.8 %. We omit the per shift results from Table 2 since it adds no additional insight. This method is consistently better than using the source only **svm**, but worse than all other adaptation methods. Note, GMA was created for a multi-view scenario where you have instance level constraints. Therefore, it is understandable that with only category level constraints this method does not perform as well as the other adaptation methods developed for use with category only constraints.

9.2 Asymmetric Features

Next, we analyze the effectiveness of our asymmetric transform learning methods by experimenting with the setting when source and target have different feature dimensions. We use the same experimental setup as previously, but use the full 31 category *Office* dataset and an alternate representation for the *dslr* domain, which is SURF BoW quantized to 600 dimensions (denoted as *dslr-600*). We compare our **mmdt** and **arc-t** methods against $\mathbf{svm}_t$ and **hfa**. Note that our **symm** method and some baseline methods ($\mathbf{svm}_s$, **gfk**) are not suited for the asymmetric feature case, as they assume a consistent feature representation across domains. The results are shown in Table 3. Again, we find that our **mmdt** method can effectively learn a feature representation for the target domain that optimizes a classification objective. Our **arc-t** method has lower accuracy on this task than **mmdt**, but these results show that it still effectively leverages the source domain data by achieving much higher accuracy than the $\mathbf{svm}_t$ baseline which ignores the source domain.

9.3 Novel Categories

We next consider the setting of practical importance where labeled target examples are not available for all objects. Recall that this is a setting that many category specific adaptation methods cannot generalize to, including **hfa** (Duan et al. 2012a) and our **symm** method. Therefore, we compare results from our **mmdt** and **arc-t** methods, which learn category independent feature transforms, to the **gfk** method of Gong et al. (2012), which learns a category independent kernel to compare the domains. We use the full *Office* dataset and allow 20 labeled examples per category in the source for *amazon* and 10 labeled examples for the first 15 object categories in the target (*dslr*). For the *webcam* → *dslr* shift, we use 8 labeled examples per category in the source for *webcam* and 4 labeled examples for the first 15 object categories in the target *dslr*.

Table 3 Multi-class accuracy results on the standard supervised domain adaptation task with different feature dimensions in the source and target

source	target	$\mathbf{svm}_t$	**hfa**	**arc-t**	**mmdt**
amazon	dslr-600	52.9 ± 0.7	57.8 ± 0.6	58.2 ± 0.6	**62.3 ± 0.8**
webcam	dslr-600	51.8 ± 0.6	60.0 ± 0.6	58.2 ± 0.7	**63.3 ± 0.5**

The target domain is *dslr* for both cases
Bold indicates the best performing result

Table 4 Multi-class accuracy results on the Office dataset for the domain shift of *webcam* → *dslr* for target test categories not seen at training time

source	$\mathbf{svm}_s$	gfk	arc-t	mmdt
amazon	10.3 ± 0.6	38.9 ± 0.4	41.4 ± 0.3	$\mathbf{44.6 \pm 0.3}$
webcam	51.6 ± 0.5	$\mathbf{62.9 \pm 0.5}$	59.4 ± 0.4	58.3 ± 0.5

Bold indicates the best performing result

The experimental results for the domain shift of *webcam* → *dslr* are evaluated and shown in Table 4. **mmdt** outperforms the baselines for the *amazon* → *dslr* shift and offers adaptive benefit over $\mathbf{svm}_s$ for the shift from *webcam* → *dslr*. As in the first set of experiments, both **arc-t** and **gfk** use nearest neighbor classifiers on a learned kernel which are more suitable to the *webcam* → *dslr* shift, as these two domains are initially very similar.

9.4 Large-scale Data

With our last experiment, we show that our method not only offers high accuracy performance; it also scales well with an increasing dataset size. Specifically, the number of constraints our algorithm optimizes scales linearly with the number of training points. Conversely, the number of constraints that need to be optimized for the **arc-t** baseline is quadratic in the number of training points.

To demonstrate the effect that constraint set size has on run-time performance, we perform experiments on the *Bing* (source) and *Caltech256* domains described in Sect. 8.2. The left-hand plot in Fig. 4 presents multi-class accuracy for this setup. Additionally, the training time of our method and that of the baselines is shown on the right-hand plot.

Our **mmdt** method provides a considerable improvement over **arc-t** and all the baselines in terms of multi-class accuracy. It is also considerably faster than all but the **gfk** method. Note that **hfa** and **gfk** do not vary significantly as the number of target training points increases. However, for **hfa** the main bottleneck time is consumed by a distance computation between each pair of training points. Therefore, since there are many more source training points than target, adding a few more target points does not significantly increase the overall time spent for this experiment, but would present a problem as the size of the dataset grew in general.

10 Conclusion

We have presented a unified framework for learning a category invariant transformation that has been proven effective for visual domain adaptation. In particular, we derive two specific formulations from the general framework, one which is most useful for learning a similarity function between a source and target domain independent of the classifier, and another which focuses on learning linear classifiers in a max-margin framework.

We demonstrated the importance of using a domain adaptation method to boost overall performance for visual recognition tasks, and analyze the scenarios in which a max-margin objective and a transformation-based approach are most beneficial. In our experiments, we provided an in-depth analysis and comparison of the different algorithms we presented and their connection to other state-of-the-art methods.

In the future, we would like to extend further to a multi-domain scenario, where lots of labeled and heterogenous source data can be exploited to help classification in a target domain.

Additionally, in this paper, we addressed the problem of adapting between source and target domains assuming a particular, fixed, image representation. Therefore, all approaches we discuss may be applied regardless of the representation choice. Many recent works have shown that strong image representations can be learned using convolutional neural networks (Krizhevsky et al. 2012; Chopra et al. 2013; Donahue et al. 2014). Our preliminary experiments demonstrate that domain adaptation bias is still present even with the convo-

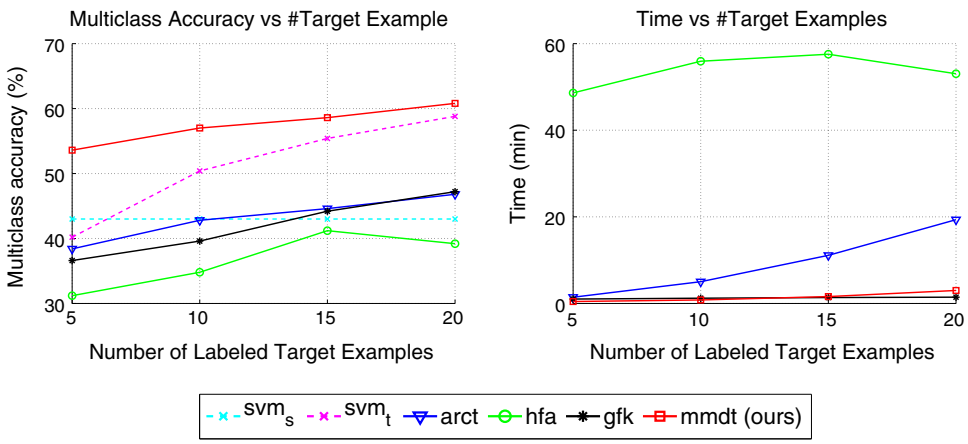


Fig. 4 *Left* multi-class accuracy on the *Bing* dataset using 50 training examples in the source and varying the number of available labeled examples in the target. *Right* training time comparison

lutional features. In the future, we would like to experiment with domain adaptation algorithms using convolutional features as well as a unified convolutional framework to learn both features and adaptation simultaneously.

Appendix: Proofs

Proof of Lemma 1 Let $\boldsymbol{W}$ have singular value decomposition $\boldsymbol{U}\boldsymbol{\Sigma}\tilde{\boldsymbol{U}}^T$. We can therefore write $\boldsymbol{W}$ as $\boldsymbol{W} = \sum_{j=1}^{p} \sigma_j \boldsymbol{u}_j \tilde{\boldsymbol{u}}_j^T$, where p is the rank of $\boldsymbol{W}$.

Claim: $\boldsymbol{u_j} \in \mathcal{C}(\mathbf{X}), \tilde{\boldsymbol{u_j}} \in \mathcal{C}(\mathbf{Z})$ such that there exists vectors, $\boldsymbol{v_j}, \tilde{\boldsymbol{v_j}}$ where $\boldsymbol{u_j} = \mathbf{X}\boldsymbol{v_j}$ and $\tilde{\boldsymbol{u_j}} = \mathbf{Z}\tilde{\boldsymbol{v_j}}$.

Proof: Let us consider what would happen if this were not true. By definition if a vector is not in the column space of the matrix then it is in the left-null space of that matrix. Namely, if $\boldsymbol{u_j} \notin \mathcal{C}(\mathbf{X})$ then $\mathbf{X}^T\boldsymbol{u_j} = 0$ and similarly if $\tilde{\boldsymbol{u_j}} \notin \mathcal{C}(\mathbf{Z})$ then $\mathbf{Z}^T\tilde{\boldsymbol{u_j}} = 0$. Now, consider that the constraints in the optimization problem we are solving only consider $\boldsymbol{W}$ in terms of the similarity function $\text{sim}(\boldsymbol{W}, \mathbf{X}, \mathbf{Z}) = \mathbf{X}^T\boldsymbol{W}\mathbf{Z} = \sum_{j=1}^{p} \sigma_j \mathbf{X}^T\boldsymbol{u_j}\tilde{\boldsymbol{u_j}}^T\mathbf{Z}$. If either $\boldsymbol{u_j} \notin \mathcal{C}(\mathbf{X})$ or $\tilde{\boldsymbol{u_j}} \notin \mathcal{C}(\mathbf{Z})$ then the corresponding element in the sum would equal zero (since the global minimizer of each $r_j(\sigma_j)$ is assumed to be zero) and the j^{th} singular value would be left unconstrained and hence automatically set to zero by the regularizer. Therefore, if a singular value $\sigma_j \neq 0$ then we know that the corresponding singular vectors are in the column space of source and target data.

Following the above claim, let $\boldsymbol{v_j}, \tilde{\boldsymbol{v_j}}$ be the vectors such that $\boldsymbol{u_j} = \mathbf{X}\boldsymbol{v_j}$ and $\tilde{\boldsymbol{u_j}} = \mathbf{Z}\tilde{\boldsymbol{v_j}}$. Then we can re-write $\boldsymbol{W}$ as follows:

$$\begin{aligned}\boldsymbol{W} &= \sum_{j=1}^{t} \sigma_j \boldsymbol{u}_j \tilde{\boldsymbol{u}}_j^T = \sum_{j=1}^{t} \sigma_j \mathbf{X}\boldsymbol{v}_j \tilde{\boldsymbol{v}}_j^T \mathbf{Z}^T \\ &= \mathbf{X}\Big(\sum_{j=1}^{t} \sigma_j \boldsymbol{v}_j \tilde{\boldsymbol{v}}_j^T\Big)\mathbf{Z}^T = \mathbf{X}\tilde{\boldsymbol{L}}\mathbf{Z}^T,\end{aligned}$$

where $\tilde{\boldsymbol{L}} = \sum_{j=1}^{t} \sigma_j \boldsymbol{v}_j \tilde{\boldsymbol{v}}_j^T$. With the transformation $\boldsymbol{L} = \boldsymbol{K}_{\mathcal{X}}^{1/2}\tilde{\boldsymbol{L}}\boldsymbol{K}_{\mathcal{Z}}^{1/2}$, we can equivalently write $\boldsymbol{W} = \mathbf{X}\boldsymbol{K}_{\mathcal{X}}^{-1/2}\boldsymbol{L}\boldsymbol{K}_{\mathcal{Z}}^{-1/2}\mathbf{Z}^T$, which proves the lemma and will simplify the theorem proof.

Proof of Theorem 1 Denote $\boldsymbol{V}_{\mathcal{X}} = \mathbf{X}\boldsymbol{K}_{\mathcal{X}}^{-1/2}$ and $\boldsymbol{V}_{\mathcal{Z}} = \mathbf{Z}\boldsymbol{K}_{\mathcal{Z}}^{-1/2}$. Note that $\boldsymbol{V}_{\mathcal{X}}$ and $\boldsymbol{V}_{\mathcal{Z}}$ are orthogonal matrices. From the lemma, $\boldsymbol{W} = \boldsymbol{V}_{\mathcal{X}}\boldsymbol{L}\boldsymbol{V}_{\mathcal{Z}}^T$; let $\boldsymbol{V}_{\mathcal{X}}^{\perp}$ and $\boldsymbol{V}_{\mathcal{Z}}^{\perp}$ be the orthogonal complements to $\boldsymbol{V}_{\mathcal{X}}$ and $\boldsymbol{V}_{\mathcal{Z}}$, and let $\bar{\boldsymbol{V}}_{\mathcal{X}} = [\boldsymbol{V}_{\mathcal{X}} \;\; \boldsymbol{V}_{\mathcal{X}}^{\perp}]$ and $\bar{\boldsymbol{V}}_{\mathcal{Z}} = [\boldsymbol{V}_{\mathcal{Z}} \;\; \boldsymbol{V}_{\mathcal{Z}}^{\perp}]$. Then

$$\begin{aligned}r\left(\bar{\boldsymbol{V}}_{\mathcal{X}}\begin{bmatrix}\boldsymbol{L} & 0\\ 0 & 0\end{bmatrix}\bar{\boldsymbol{V}}_{\mathcal{Z}}^T\right) &= r\left(\begin{bmatrix}\boldsymbol{W} & 0\\ 0 & 0\end{bmatrix}\right) = r(\boldsymbol{W}) + r(0) \\ &= r(\boldsymbol{W}) + \text{const}.\end{aligned}$$

One can easily verify that, given two orthogonal matrices $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ and an arbitrary matrix $\boldsymbol{M}$, $r(\boldsymbol{V}_1\boldsymbol{M}\boldsymbol{V}_2) = \sum_j r_j(\sigma_j)$ if σ_j are the singular values of $\boldsymbol{M}$. So

$$r\left(\bar{\boldsymbol{V}}_{\mathcal{X}}\begin{bmatrix}\boldsymbol{L} & 0\\ 0 & 0\end{bmatrix}\bar{\boldsymbol{V}}_{\mathcal{Z}}^T\right) = \sum_j r_j(\bar{\sigma_j}) + \text{const} = r(\boldsymbol{L}) + \text{const},$$

where $\bar{\sigma_i}$ are the singular values of $\boldsymbol{L}$. Thus, $r(\boldsymbol{W}) = r(\boldsymbol{L}) + \text{const}$.

Finally, rewrite the similarity values using the previously derived kernel representation of the transformation matrix $\boldsymbol{W} = \mathbf{X}\boldsymbol{K}_{\mathcal{X}}^{-1/2}\boldsymbol{L}\boldsymbol{K}_{\mathcal{Z}}^{-1/2}\mathbf{Z}^T$:

$$\begin{aligned}\text{sim}(\boldsymbol{W}, \mathbf{X}, \mathbf{Z}) &= \mathbf{X}^T\boldsymbol{W}\mathbf{Z} = \boldsymbol{K}_{\mathcal{X}}\boldsymbol{K}_{\mathcal{X}}^{-1/2}\boldsymbol{L}\boldsymbol{K}_{\mathcal{Z}}^{-1/2}\boldsymbol{K}_{\mathcal{Z}} \\ &= \boldsymbol{K}_{\mathcal{X}}^{1/2}\boldsymbol{L}\boldsymbol{K}_{\mathcal{Z}}^{1/2} = \text{sim}(\boldsymbol{L}, \boldsymbol{K}_{\mathcal{X}}^{1/2^T}, \boldsymbol{K}_{\mathcal{Z}}^{1/2})\end{aligned}$$

The theorem follows by rewriting r and the constraints $c_{\boldsymbol{W}}$ using the above derivations in terms of $\boldsymbol{L}$. Note that both $r(\boldsymbol{W})$ and the $c_{\boldsymbol{W}}$ can be computed independently of the dimension of $\boldsymbol{W}$, so simple arguments show that the optimization may be solved in polynomial time independent of the dimension when the r_j functions are convex.

References

Argyriou, A., Micchelli, C. A., & Pontil, M. (2010). On spectral learning. *Journal of Machine Learning Research*, *11*, 935–953.

Aytar, Y., & Zisserman, A. (2011). Tabula rasa: Model transfer for object category detection. In *Proceedings of the international conference on computer vision (ICCV)* (pp. 2252–2259).

Ben-david, S., Blitzer, J., Crammer, K., & Pereira, O. (2007). Analysis of representations for domain adaptation. In *Advances in neural information processing systems (NIPS)* (pp. 137–145). Cambridge: MIT Press.

Bergamo, A., & Torresani, L. (2010). Exploiting weakly-labeled web images to improve object classification: A domain adaptation approach. In *Advances in neural information processing systems (NIPS)* (pp. 181–189).

Blitzer, J., Dredze, M., & Pereira, F. (2007). Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. *ACL*, *7*, 440–447.

Chopra, S., Balakrishnan, S., & Gopalan, R. (2013). Dlid: Deep learning for domain adaptation by interpolating between domains. In *ICML workshop on challenges in representation learning*.

Dai, W., Chen, Y., Xue, G., Yang, Q., & Yu, Y. (2008). Translated learning: Transfer learning across different feature spaces. In *Advances in neural information processing systems (NIPS)* (pp. 353–360).

Daume III, H. (2007). Frustratingly easy domain adaptation. In *ACL* (pp. 256–263).

Davis, J., Kulis, B., Jain, P., Sra, S., & Dhillon, I. (2007). Information-theoretic metric learning. In *Proceedings of the international conference on Machine learning (ICML)* (pp. 209–216) .

Diethe, T., Hardoon, D. R., & Shawe-Taylor, J. (2010). Constructing nonlinear discriminants from multiple data views. In *Machine learning and knowledge discovery in databases* (pp. 328–343) Berlin: Springer.

Donahue, J., Hoffman, J., Rodner, E., Saenko, K., & Darrell, T. (2013). Semi-supervised domain adaptation with instance constraints. In

Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 668–675).

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., et al. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference in machine learning (ICML)*.

Duan, L., Tsang, I. W., Xu, D., & Maybank, S. J. (2009). Domain transfer svm for video concept detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1375–1381).

Duan, L., Xu, D., & Tsang, I. W. (2012a). Learning with augmented features for heterogeneous domain adaptation. In *Proceedings of the international conference on machine learning* (pp. 711–718).

Duan, L., Xu, D., Tsang, I. W. H., & Luo, J. (2012b). Visual event recognition in videos by learning from web data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(9), 1667–1680.

Farhadi, A., & Tabrizi, M. K. (2008). Learning to recognize activities from the wrong view point. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 154–166).

Farquhar, J., Hardoon, D., Meng, H., Shawe-taylor, J. S., & Szedmak, S. (2005). Two view learning: Svm-2k, theory and practice. In *Advances in neural information processing systems (NIPS)* (pp. 355–362).

Gong, B., Shi, Y., Sha, F., & Grauman, K. (2012). Geodesic flow kernel for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2066–2073).

Gopalan, R., Li, R., & Chellappa, R. (2011). Domain adaptation for object recognition: An unsupervised approach. In *Proceedings of the international conference on computer vision (ICCV)* (pp. 999–1006).

Hoffman, J., Rodner, E., Donahue, J., Saenko, K., & Darrell, T. (2013). Efficient learning of domain-invariant image representations. In *International conference on learning representations (ICLR)*. http://arxiv.org/abs/1301.3224

Jhuo, I. H., Liu, D., Chang, S. F., & Lee, D. T. (2012). Robust visual domain adaptation with low-rank reconstruction. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2168–2175).

Jiang, J. (2008). A literature survey on domain adaptation of statistical classifiers. http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/.

Jiang, J., & Zhai, C. X. (2007). Instance weighting for domain adaptation in NLP. In *ACL* (pp. 264–271).

Jiang, W., Zavesky, E., Chang, S., & Loui, A. (2008). Cross-domain learning methods for high-level visual concept classification. In *International conference on image processing (ICIP)* (pp. 161–164).

Kan, M., Shan, S., Zhang, H., Lao, S., & Chen, X. (2012). Multi-view discriminant analysis. In *Proceedings of the European computer vision conference (ECCV)* (pp. 808–821). Berlin: Springer.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*.

Kulis, B., Jain, P., & Grauman, K. (2009). Fast similarity search for learned metrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(12), 2143–2157.

Kulis, B., Saenko, K., & Darrell, T. (2011). What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1785–1792).

Li, R., & Zickler, T. (2012). Discriminative virtual views for cross-view action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2855–2862).

Li, X. (2007). Regularized adaptation: Theory, algorithms and applications. Ph.D. thesis, USA: University of Washington

Quadrianto, N., & Lampert, C. H. (2011). Learning multi-view neighborhood preserving projections. In *Proceedings of the International Conference on Machine Learning (ICML)* (pp. 425–432).

Rodner, E., Hoffman, J., Donahue, J., Darrell, T., Saenko, K. (2013). Towards adapting imagenet to reality: Scalable domain adaptation with implicit low-rank transformations. arXiv:1308.4200 (preprint).

Saenko, K., Kulis, B., Fritz, M., & Darrell, T. (2010). Adapting visual category models to new domains. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 213–226).

Sharma, A., Kumar, A., Daume, H., & Jacobs, D. W. (2012). Generalized multiview analysis: A discriminative latent space. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2160–2167).

Torralba, A., & Efros, A. (2011). Unbiased look at dataset bias. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1521–1528).

Yang, J., Yan, R., & Hauptmann, A. G. (2007). Cross-domain video concept detection using adaptive svms. In *ACM Multimedia* (pp 188–197).