What You Saw is Not What You Get: Domain Adaptation Using Asymmetric Kernel Transforms

Brian Kulis, Kate Saenko, and Trevor Darrell
UC Berkeley EECS and ICSI
{kulis,saenko,trevor}@eecs.berkeley.edu

Abstract

In real-world applications, "what you saw" during training is often not "what you get" during deployment: the distribution and even the type and dimensionality of features can change from one dataset to the next. In this paper, we address the problem of visual domain adaptation for transferring object models from one dataset or visual domain to another. We introduce ARC-t, a flexible model for supervised learning of non-linear transformations between domains. Our method is based on a novel theoretical result demonstrating that such transformations can be learned in kernel space. Unlike existing work, our model is not restricted to symmetric transformations, nor to features of the same type and dimensionality, making it applicable to a significantly wider set of adaptation scenarios than previous methods. Furthermore, the method can be applied to categories that were not available during training. We demonstrate the ability of our method to adapt object recognition models under a variety of situations, such as differing imaging conditions, feature types and codebooks.

1. Introduction

The vast majority of object recognition methods are evaluated on the same dataset as the one they were trained on. However, each image dataset corresponds to a particular "visual domain" with its own peculiarities: compare *amazon.com* product images to consumer snapshots of the same product in Figure 1. There is substantial evidence that standard classification models degrade significantly when presented with test points from a different domain (for example, see [9] for a discussion focused on natural language processing). Recently, there has been increasing interest in understanding and overcoming the *visual domain adaptation* problem: given a target image domain whose feature distribution is different from that of a given source domain, how can we effectively utilize models learned on the source domain at test time?

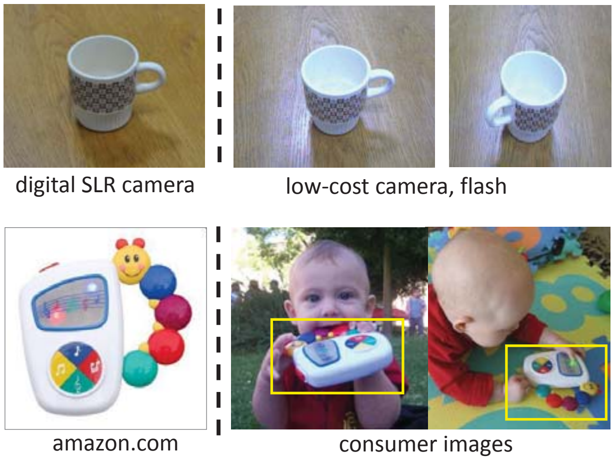
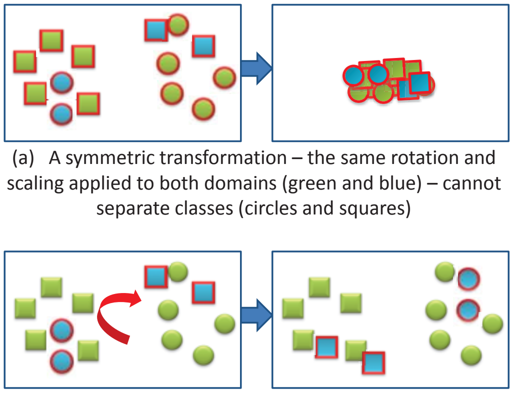


Figure 1. We address the problem of adapting object models trained on a particular source dataset, or domain (left), to a target domain (right).

Recently, the work of [19, 21, 12] examined the domain adaptation problem for computer vision tasks, such as video concept detection and visual object modeling. In particular, [19] learned a domain-invariant distance metric using a small number of labeled images in the target domain. However, these proposed methods, as well as nearly all other methods presented outside the vision community for domain adaptation, assume that the underlying representations of the domains share the same feature space, with the same dimensionality. For example, one baseline for adaptation methods is to take a classification model from the source domain (say, an SVM) and apply it directly to the target domain, which is impossible if the domains have different representations. As a result, thus far there are very few methods for performing adaptation for scenarios including different image representations.

In this paper, we introduce a novel domain adaptation technique based on learning cross-domain transformations. The key idea is to learn an asymmetric non-linear transformation that maps points from one domain to another do-



(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

Figure 2. A conceptual illustration of how an asymmetric domain transform (this paper) can be more flexible than a symmetric one [19].

main using supervised data from both domains. The ability to learn asymmetric and non-linear transformations is key to our approach, as it allows us to handle more general types of domain shift and changes in feature type and dimension. The input to the algorithm consists of pairs of inter-domain examples that are known to be semantically similar (or dissimilar). We present a general model for learning linear cross-domain transformations, and then prove a novel result showing how to learn non-linear transformations by kernelizing the formulation for a particular class of regularizers. We show that the method of [19] is a special case of our general formulation, producing symmetric positive definite transformations, and argue that asymmetric indefinite transformations are more flexible for a variety of adaptation tasks (see Figure 2 for a motivating example). Encoding the domain invariance into the feature representation allows our method to benefit from a broad range of classification methods, from k-NN to SVM, as well as clustering methods.

Importantly, our approach can be applied to the scenario where some of the categories do not have any labels in the target domain, essentially transferring the learned "domain shift" to new categories encountered in the target domain. Thus, it can be thought of as a form of knowledge transfer from the source to the target domain. However, in contrast to many existing transfer learning paradigms (e.g. [20], [14]), we transfer the structure of the domain shift, including changes in representation, rather than structures common to related categories.

In the next section, we relate our approach to existing work on domain adaptation. Section 3 describes the theoretical framework behind our approach, including novel results on the possibility of kernelization of the asymmetric transform, and presents the main algorithm. We evaluate our method on a dataset designed to study the problem of visual domain shift, and show results of object classifier adaptation on several challenging shifts in Section 4.

2. Related Work

Learning transformations has been an important problem in both the vision and machine learning communities (see [19, 7, 15, 16, 6] for some vision examples). To our knowledge, the only existing method applied to visual category adaptation is the method of [19]. This method learns a metric to compare two cross-domain data points that satisfies a set of given cross-domain (dis)similarity constraints. Because the Mahalanobis distance is used for learning, the algorithm essentially treats both the source and target domains as part of a single data set, and applies existing metric learning methods to learn a transformation over this data. We will argue that this approach is insufficient when the dimensionalities of the domains are different, and restricts the type of transformations that can be learned.

Other vision approaches for cross-domain transfer include SVM-based methods: The method of [21] proposed an adaptive SVM, 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. Domain transfer SVM [11] 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 [12] utilizes adaptive multiple kernel learning to learn a kernel function based on multiple base kernels. The disadvantage of [21, 11, 12] is the inability to transfer the adapted function to novel categories, which is limiting in object recognition scenarios, where the set of available category labels varies among datasets. Other authors have proposed "translating" features between domains; [13] translated features between camera views to transfer activity models, while [8] translated user preferences between text and image domains.

Most of the work on adaptation has been focused outside of the vision community (also see [18] for a discussion of the more general dataset shift problem). One of the prominent approaches was proposed by Daume [9], who introduced a feature replication method for domain adaptation. The basic idea is that, given a feature vector $\boldsymbol{x}$, we define the augmented feature vector $\tilde{\boldsymbol{x}} = (\boldsymbol{x}; \boldsymbol{x}; \boldsymbol{0})$ for data points in the source and $\tilde{\boldsymbol{x}} = (\boldsymbol{x}; \boldsymbol{0}; \boldsymbol{x})$ for data points in the target. Daume also gives an overview of the relevant baselines, which we employ in this work. Structural correspondence learning is another method proposed for NLP tasks such as sentiment classification [4]. This method relies on so-called pivot features—words that frequently occur in both domains and are correlated with domain-specific words—so it is not clear how such an approach could easily be used for visual category adaptation tasks.

We stress that nearly all existing methods, including ones in vision [11, 12, 21] as well as ones outside vision [9], cannot be applied when the underlying dimensionality of the source and target domain are different.

3. Domain Adaptation Using Regularized Cross-Domain Transforms

In this section, we present our adaptation approach, contrasting with the work of [19]. We present a novel kernelization result which will allow us to learn non-linear transformations, and use this to design an algorithm for learning general cross-domain transformations.

Denote the source domain as $\mathcal{A}$, with data points $\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_A}$, and the target domain as $\mathcal{B}$, with data points $\boldsymbol{y}_1, ..., \boldsymbol{y}_{n_B}$. In general, the dimensionality of the source domain points d_A will be different from the dimensionality of the target domain points d_B.

3.1. Background: Symmetric Transforms

The method of Saenko et al. [19] proposes to model the adaptation problem using Information-theoretic Metric Learning (ITML), which can be viewed as learning a linear transformation W between $\mathcal{A}$ and $\mathcal{B}$, optimised to satisfy constraints between transformed points, which are expressed as functions of $\boldsymbol{x}^T W \boldsymbol{y}$, where $\boldsymbol{x} \in \mathcal{A}$ and $\boldsymbol{y} \in \mathcal{B}$. The authors apply their method in kernel space in order to learn non-linear transformations, using known kernelization results. While this model is intuitively appealing for domain adaptation, the authors of [19] make a key simplifying assumption that weakens the model but simplifies the algorithm: they choose the *LogDet regularizer* over W [16], which restricts the data *only* to the case when $d_A = d_B$ since the matrix trace and determinant are only defined over square matrices W. Another important restriction of the LogDet regularizer is that it is only defined over symmetric positive definite matrices W. Positive definiteness implies that the method learns a global transformation that is applied to *both* domains (as in Figure 2a), and so LogDet forces yet another restriction on the transformation model.

3.2. Asymmetric Transforms

Our goal is to extend the model of [19] to the more general case where the domains are not restricted to be the same dimensionality, and arbitrary asymmetric transformations can be learned. In order to avoid the restrictions of the ITML model for adaptation, we seek an alternative regularizer that can generalize the model to use domains of differing dimensionalities but still retains the benefits of kernelization.

The objective function proposed by [19] for finding the linear transformation matrix W (in the unconstrained case) may be expressed in generality as:

$$\min_W r(W) + \lambda \sum_i c_i(X^T W Y), \tag{1}$$

where X is the matrix of all the points in $\mathcal{A}$, Y is the matrix of all points in $\mathcal{B}$, r is a matrix regularizer, and the c_i are the loss functions over the constraints, assumed to be expressed as a function of the matrix $X^T W Y$. Thus, the optimization aims to minimize a matrix regularizer r plus a set of constraints that are a function of the learned inner products $\boldsymbol{x}^T W \boldsymbol{y}$. As an example, a possible constraint could be that $\boldsymbol{x}^T W \boldsymbol{y}$ be close to some target value t_i as encoded by the constraint $(\boldsymbol{x}^T W \boldsymbol{y} - t_i)^2$.

We focus on the particular regularizer $r(W) = \frac{1}{2}\|W\|_F^2$ and constraints that are a function of $\text{sim}_W(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T W \boldsymbol{y}$ for similar or dissimilar $\boldsymbol{x}, \boldsymbol{y}$ pairs. 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. However, in showing kernelization for this regularizer, we will 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 Section 3.2.1. Equipped with this result, we then describe our overall algorithm in detail in Section 3.2.2.

3.2.1 Kernelization Analysis

There are two main limitations to the transformation learning problem (1) presented above. First, it is limited to linear transformations W, which may not be sufficient for some adaptation tasks. Second, the size of W grows as $d_A \cdot d_B$, which may be prohibitively large for some problems. In this section, we prove that (1) may be solved in kernel space *for a wide class of regularizers*, resulting in non-linear transformations whose complexity is independent of the dimensionalities of the points in either domain. This kernelization result is the first general kernelization result for asymmetric transformation learning, and is critical to obtaining good performance for several domain adaptation tasks. Note that kernelization has been proven for some metric learning formulations, such as [16]; in all these cases, the kernelization results assume that W is symmetric positive definite, whereas our results hold for arbitrary W. We also note connections to the work of [1], 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 i) the learned similarity function resulting from solving (1) can be computed only using inner products between data points in $\mathcal{A}$ and inner products between data points in $\mathcal{B}$, and

ii) (1) can be reformulated as an optimization problem involving such inner products and whose size is independent of the dimensionalities d_A and d_B. Then we can replace standard inner products with arbitrary kernel functions such as the Gaussian RBF kernel function, resulting in nonlinear learned transformations between the input space of the source domain and the input space of the target domain. Our analysis will hold for the class of regularizers $r(W)$ that can be written in terms of the singular values of W; that is, if $\sigma_1, ..., \sigma_p$ are the singular values of W, then $r(W)$ is of the form $\sum_{j=1}^{p} r_j(\sigma_j)$ for some scalar functions r_j. For example, the squared Frobenius norm $r(W) = \frac{1}{2}\|W\|_F^2$ is a special case where $r_j(\sigma_j) = \frac{1}{2}\sigma_j^2$. In the following analysis, the input kernel matrices over within-domain points are given as $K_A = X^T X$ and $K_B = Y^T Y$. We begin with the first result (proof in Appendix A).

Lemma 3.1. *Assume that K_A and K_B are strictly positive definite. Further assume that the regularizer r is convex, $r(W)$ is of the form $\sum_j r_j(\sigma_j)$, where $\sigma_1, ..., \sigma_p$ are the singular values of W, and that the global minimizer of each r_j is 0. Then there exists an $n_A \times n_B$ matrix L such that the optimal solution W^* to (1) is of the form $W^* = XK_A^{-1/2}LK_B^{-1/2}Y^T$.*

Note that the assumption that the global minimizer of each r_j is 0 can be eliminated in some cases, but we leave out the discussion of this more general case due to space constraints. One important consequence of the above lemma is that, given arbitrary points $\boldsymbol{x}$ and $\boldsymbol{y}$, the function $\text{sim}_W(\boldsymbol{x}, \boldsymbol{y})$ can be computed in kernel space—by replacing W with $XK_A^{-1/2}LK_B^{-1/2}Y^T$, the expression $\boldsymbol{x}^T W \boldsymbol{y}$ can be written purely in terms of inner products.

The above result demonstrates the existence of such a matrix L, but does not show how to compute it. Using the above lemma, we now show how to equivalently rewrite the optimization (1) in terms of the kernel matrices K_A and K_B to solve for L (proof in Appendix A):

Theorem 3.2. *Assume the conditions of Lemma 3.1 hold. If W^* is the optimal solution to (1) and L^* is the optimal solution to the following problem:*

$$\min_L r(L) + \lambda \sum_i c_i(K_A^{1/2} L K_B^{1/2}), \tag{2}$$

then $W^* = XK_A^{-1/2}L^*K_B^{-1/2}Y^T$.

To summarize, the main theorem demonstrates that, instead of solving (1) for W directly, we can *equivalently* solve (2) for L, and then implicitly construct W via $W = XK_A^{-1/2}LK_B^{-1/2}Y^T$. In particular, this form of W allows us to compute $\boldsymbol{x}^T W \boldsymbol{y}$ using only kernel functions. Though our focus on this paper is on one particular regularizer—the squared Frobenius norm—one can imagine applying our analysis to other regularizers. For example, the trace norm $r(W) = \text{tr}(W)$ also falls under our framework; because the trace-norm as a regularizer is known to produce low-rank matrices W, it would be desirable in kernel dimensionality-reduction settings. We leave the study of the trace-norm regularizer for domain adaptation as potential future work.

3.2.2 Setup, Constraint Generation, and Optimization

We now detail the training and test setup, the generation of constraints, and our optimization method.

We consider two scenarios in our experiments. In the first scenario, all categories are present at training. In this case, we assume that we have many labeled examples for each category in the source domain, and a few labeled examples for each category in the target domain. We form the following class-based constraints for each pair of training points $(\boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{x} \in \mathcal{A}$ and $\boldsymbol{y} \in \mathcal{B}$: if $\boldsymbol{x}$ and $\boldsymbol{y}$ are from the same category, we construct the constraint

$$c_i(X^T W Y) = (\max(0, \ell - \boldsymbol{x}^T W \boldsymbol{y}))^2,$$

and if $\boldsymbol{x}$ and $\boldsymbol{y}$ are in different categories, we construct the constraint

$$c_i(X^T W Y) = (\max(0, \boldsymbol{x}^T W \boldsymbol{y} - u))^2.$$

Note that, in both cases, the constraints are a function of $X^T W Y$ since $\boldsymbol{x}^T W \boldsymbol{y}$ is simply a single entry of the matrix $X^T W Y$. Here ℓ and u are lower and upper bound parameters chosen so that for same-category pairs, the learned similarity $\boldsymbol{x}^T W \boldsymbol{y}$ should be large, and for different-category pairs, the learned similarity $\boldsymbol{x}^T W \boldsymbol{y}$ should be small. Note the difference here between our approach and [19]: we constrain based on the similarity function, as opposed to the Mahalanobis distance used in [19].

In the second scenario, only a subset of categories are present at training time. As in the first scenario, we train our model on the available training data from the source and target by constructing class-based constraints. However, at test time, we aim to adapt the classifier learned over the training categories to a set of new categories, for which we only have labels in the source domain. We can achieve this by applying the learned transformation to map target data to the source domain, and apply a classifier such as k-nearest neighbors from the mapped test data to the labeled source data.

Following [19], we can also generate constraints based not on class labels, but on other similarity or dissimilarity information that may be available. For example, if the source and target data include images of the same object instances, and we have such information available, we can still learn about the structure of the domain shift without needing class label information. Such constraints are called *correspondence* constraints.

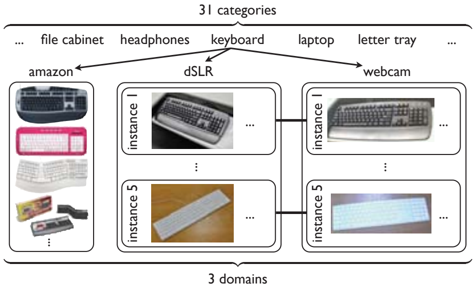


Figure 3. Domain adaptation dataset for investigating domain shifts in visual category recognition tasks (figure is from [19]).

We conclude with a few remarks regarding learning in kernel space and our optimization method. First, we note that when mapping to kernel space, (2) indicates that the constraint $c_i(X^TWY)$ is mapped to $c_i(K_A^{1/2}LK_B^{1/2})$; for a similarity constraint over a pair $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$, this mapping is given by $(\max(0, (\ell - \boldsymbol{e}_i^T K_A^{1/2} L K_B^{1/2} \boldsymbol{e}_j))^2$ for same-class pairs (and analogously for different-class pairs). Second, we form the input kernel matrices K_A and K_B using a Gaussian RBF kernel, unless otherwise stated. Third, the regularizer considered in our paper ($r(W) = \frac{1}{2}\|W\|_F^2$) is strictly convex, and one can therefore use a variety of possible optimization techniques for minimizing (2). We opted for an alternating projection method based on Bregman's algorithm. This method updates the transformation with respect to a single constraint c_i at a time. It can be easily implemented to scale to large problems and has fast convergence in practice. See Censor and Zenios for details on Bregman's algorithm [5]. Alternately, one could use a simple gradient descent or stochastic gradient descent procedure over (2).

4. Experiments

In this section, we evaluate our domain adaptation approach, **ARC-t**, by applying it to classification of object categories using nearest neighbor classifiers.

The data set used in our experiments is the domain adaptation benchmark introduced in [19]. This data set contains images from 31 object categories and three image domains. The first domain, amazon, contains product images, typically in a canonical pose with a white (or uniform) background. The second domain, dslr, contains images of the same object categories taken with a digital SLR camera in an office. These images are high-resolution with varying poses and backgrounds. Finally, the webcam domain contains images of the same objects as the dslr domain, but taken with a webcam using a flash, which are of low-resolution with varying poses and backgrounds. Note that in our experiments, images of the same test object are held out of training. See Figure 3, or [19] for details about the data set. We create several additional domain shifts by changing the codebooks and the type of features used to compute visual words, as detailed below.

Baselines and Competing Methods: Following [9] and [19], we ran several baseline classifiers in our experiments. One of the key difficulties in running these experiments is that it is not obvious how to even build a baseline when the domains have different image features; in such cases, even a simple k-nearest neighbor classifier cannot be applied directly. As a solution, *only* for the baselines below that cannot be applied when the dimensionalities are different, we apply *kernel canonical correlation analysis* (KCCA) to project the data from both domains into a common space. Briefly, let X_A^{CCA} be a matrix of training points from the source and X_B^{CCA} be a matrix of training points from the target such that the training points align, i.e., the i-th source domain point is the same object as the i-th target domain point. KCCA finds projections U_A and U_B to maximize the cross-correlation of $U_A X_A^{CCA}$ and $U_B X_B^{CCA}$ (we choose the number of rows/components of U_A and U_B to be equal to the number of aligned training points). We can then define a baseline similarity (or kernel) between arbitary points $\boldsymbol{x}$ from the source and $\boldsymbol{y}$ from the target as $\boldsymbol{x}^T U_A^T U_B \boldsymbol{y}$. As shown previously, this entire process can be kernelized [2], and is a natural method for comparing correlated sets of variables, such as different-dimensional feature vectors extracted from similar images.

We compare our approach against the following methods. Note that these are representative of the state-of-the-art in domain adaptation. In all of the below methods, when either the image features or codebooks being compared are different, we first apply KCCA (see above).

- **knn-ab** applies k-nearest neighbors by comparing test images from the target to training images from the source.
- **knn-bb** applies k-nearest neighbors by comparing test points in the target only against the training points in the target domain.
- **ml-bb** applies metric learning [10] before the k-nn classification. In this case, the metric is trained only over the training points from the target domain, and then **knn-bb** is applied using the learned metric.
- **symm** is the method of [19], which applies a cross-domain metric, learned on all training points, before the k-nn classification.
- **ml-ab** applies the ITML algorithm trained on all training points, followed by k-nn classification, and is equivalent to the $ITML(A + B)$ baseline of [19].
- **svm-ab** applies an SVM classifier using the union of the training images in the source and target to train the SVM.
- **svm-rf** applies the domain adaptation method of [9], which replicates features to build augmented features for source and target points.

Training and Testing Details: As discussed in Section 3.2.2, we break up our experiments into two sets, following [19]. We call the first the *same-category* experiments: here, there is training data available for all categories at training time (albeit a much smaller amount in the target domain). We select 20 training images per category from the source and 3 images per category from the target. For **ARC-t**, we generate constraints across all cross-domain pairs, as described earlier, and directly follow the approach of [19] for **symm**. The second set of experiments is denoted the *new-category* experiments. Here, we break up the data such that half of the categories are used for training and half for testing. Note that, in this case, the SVM baselines **svm-ab** and **svm-rf** are not applicable, since the SVM classifier can only be applied when all categories are present at training. Thus, all baselines are based on k-nn classification. For these experiments, we select 20 training images per category from the source and 10 images per category from the target. For the new-category experiments, we additionally employ correspondence constraints based on the object instance labels for both **ARC-t** and **symm**; such labels are also used for building the KCCA embeddings in both same-category and new-category experiments.

We use an RBF kernel for **symm**, **ARC-t**, and the SVM methods, with width $\sigma = 1$. In our results, we validate the parameter λ (and the analogous learning rate parameter from SVM) over the training data, choosing from $\{10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3, 10^4\}$. The results presented are averaged over 10 runs, and we measure classification accuracy for each method. In the new-category experiments, we further test **ARC-t linear**—our method applied with a linear kernel instead of an RBF—to compare the difference between a linear and non-linear transformation.

Image Processing: All images were resized to the same width and converted to grayscale. Two types of local scale-invariant interest points were detected: SURF [3] and SIFT [17] features. Both type of features have been shown to be highly repeatable and robust to noise, displacement, geometric and photometric transformations. For SURF, the blob response threshold was set to 1000, and the other parameters to default values. A 64-dimensional non-rotationally invariant SURF descriptor was used to describe the patch surrounding each detected interest point. For SIFT, we used a Harris-affine detector and extracted a 128-dimensional descriptor. After extracting a set of descriptors for each image, vector quantization into visual words was performed to generate the final feature vector. To investigate domain shift due to a change in the vector quantization step, two different codebooks were used for SURF features: 1) a codebook of size 800, constructed by k-means clustering part of the amazon database, and 2) a codebook of size 600, constructed on the dSLR database. For SIFT features, the codebook was 900-dimentional. All images were converted to histograms over the resulting visual words. No spatial or color information was included in the image representation for these experiments.

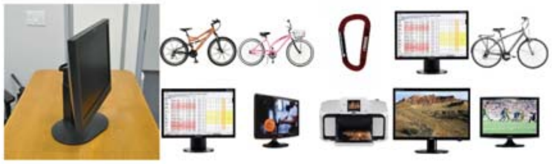


Figure 4. Examples of the 5 nearest neighbors retrieved for a dslr query image (left image) from the amazon dataset, using the non-adapted **knn-ab** baseline in Table 1 (top row of smaller images) and the learned cross-domain **ARC-t** kernel (bottom row).

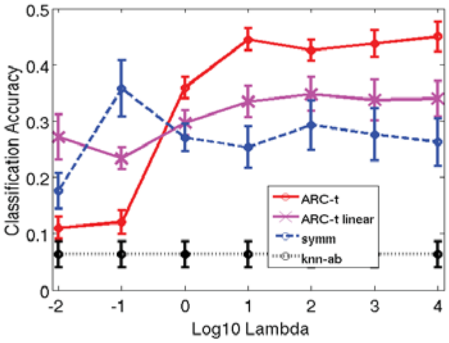


Figure 5. Plot of classification accuracy as a function of the learning rate lambda over the webcam800-dslr600 new categories experiment. This plot also shows a comparison between learning a linear transformation (ARC-t linear) and a non-linear transformation (ARC-t).

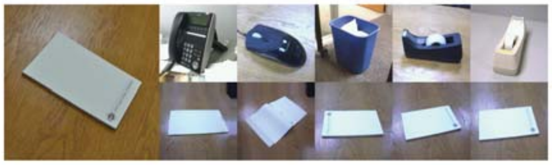


Figure 6. Examples of the 5 nearest neighbors retrieved for a dslr query image (left image) from the webcam dataset, using the **knn-ab** baseline with KCCA in Table 2 (top row of smaller images) and the learned cross-domain **ARC-t** kernel (bottom row). In this case, the codebooks of the two domains are different; it is clear that KCCA is unable to learn an effective mapping for this task.

Results: The same-category results are presented in Table 1. In general, most baselines and existing methods perform nearly as well as **ARC-t** (particularly when the features and codebooks of the domains align, as in webcam-dslr and dslr-webcam), though our method overall achieves the best results on these experiments. In this table, we tested a variety of domain adaptation settings involving all three domains from the adaptation data set, as well as varying

		Baselines / Existing Methods							This Paper
Domain A	Domain B	**knn-ab**	**knn-bb**	**svm-ab**	**svm-rf** [9]	**symm** [19]	**ml-bb** [10]	**ml-abb** [10]	**ARC-t**
dslr	webcam	24.6	31.2	34.3	34.6	35.1	32.4	31.8	**36.1**
webcam	dslr	12.2	21.1	25.1	23.8	**27.5**	24.5	19.6	25.3
amazon	dslr	4.9	47.9	46.6	46.6	49.5	47.3	43.8	**50.4**
amazon-800	dslr-600	5.4	52.1	51.0	51.0	52.3	52.6	48.1	**53.2**
webcam-surf	dslr-sift	5.1	22.1	25.9	25.8	28.5	30.0	22.0	**30.9**
amazon-surf	dslr-sift	4.7	49.6	52.8	52.8	52.5	**53.9**	46.7	53.3

Table 1. Domain adaptation results for categories seen during training in the target domain.

		Baselines / Existing Methods		This Paper	
Domain A	Domain B	**knn-ab**	**symm** [19]	**ARC-t**	**ARC-t linear**
webcam	dslr	8.4	30.3	**37.4**	32.5
webcam-800	dslr-600	9.7	35.8	**45.0**	34.8
webcam-surf	dslr-sift	9.7	17.0	**24.8**	20.6

Table 2. Domain adaptation results for categories not seen during training in the target domain. These results also show the performance of our method using a linear kernel, and demonstrates the benefits of learning a non-linear transformation.

the features and the codebook sizes. Interestingly, these experiments seem to indicate that the feature replication svm method **svm-rf** does not perform any better than the baseline **svm-ab**, at least on this data. In Figure 4, we show some examples of nearest neighbors over the amazon-dslr experiment; given a dslr query, we retrieve the nearest neighbors from amazon using **knn-ab** and our approach. Here we see a clear qualitative advantage of **ARC-t**.

The benefits of our approach are very apparent in the new-category experiments, as given in Table 2. Here, we see a significant improvement over the relevant baselines on all of the experiments. We compared the webcam and dslr domains under three settings. The first (webcam-dslr) compares the algorithms over 800-dimensional SURF codebooks, the second employs a 600-dimensional SURF codebook for dslr, and the third employs SIFT features for dslr. Figure 5 shows a more detailed display of the results on the webcam800-dslr600 experiment, with accuracy results as a function of the lambda parameter. Figure 6 presents some example nearest neighbors retrieved in the webcam800-dslr600 experiment; these results show that the **knn-ab** method, which utilizes KCCA to build a mapping between the domains, does not learn an effective transformation. Overall, our results demonstrate a significant improvement in a variety of adaptation settings, especially for the challenging task of adapting simultaneously over new categories, features, and codebooks.

Finally, in Table 2 we additionally present results for **ARC-t linear** to show how learning a linear transformation with our method compares to the kernelized version. As is evident in the results, there is a significant improvement when using the kernelized version with a Gaussian RBF kernel, and so these results validate the kernelization analysis presented in this paper.

5. Conclusion

We presented a novel approach to learning an asymmetric, non-linear transformation for domain adaptation. Unlike existing methods, our approach can be applied to learn to adapt when the source and target domains utilize different image features or codebooks. Our main technical contribution shows how a general formulation for the transformation learning problem can be applied in kernel space, resulting in non-linear transformations. We utilized this result to design an algorithm based on squared Frobenius regularization and similarity constraints. Results show clear benefits compared to existing techniques, and validate our analysis and use of kernelization.

Acknowledgements: Supported in part by awards from the US DOD and DARPA, including contract W911NF-10-2-0059, by NSF awards IIS-0905647 and IIS-0819984, and by Toyota and Google.

References

[1] A. Argyriou, C. A. Micchelli, and M. Pontil. On spectral learning. *Journal of Machine Learning Research*, 11:935–953, 2010. 1787

[2] F. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002. 1789

[3] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *ECCV*, 2006. 1790

[4] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. *ACL*, 2007. 1786

[5] Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997. 1789

[6] G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *Pattern Recognition and Image Analysis*, 2009. 1786

[7] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proc. CVPR*, 2005. 1786

[8] W. Dai, Y. Chen, G.-R. Xue, Q. Yang, and Y. Yu. Translated learning: Transfer learning across different feature spaces. In *Proc. NIPS*, 2008. 1786

[9] H. Daume III. Frustratingly easy domain adaptation. In *ACL*, 2007. 1785, 1786, 1787, 1789, 1791

[10] J. Davis, B. Kulis, P. Jain, S. Sra, and I. Dhillon. Information-theoretic metric learning. *ICML*, 2007. 1789, 1791

[11] L. Duan, I. W. Tsang, D. Xu, and S. J. Maybank. Domain transfer svm for video concept detection. In *CVPR*, 2009. 1786, 1787

[12] L. Duan, D. Xu, I. Tsang, and J. Luo. Visual event recognition in videos by learning from web data. In *Proc. CVPR*, 2010. 1785, 1786, 1787

[13] A. Farhadi and M. K. Tabrizi. Learning to recognize activities from the wrong view point. In *ECCV*, 2008. 1786

[14] M. Fink. Object classification from a single example utilizing class relevance metrics. In *Proc. NIPS*, 2004. 1786

[15] T. Hertz, A. Bar-Hillel, and D. Weinshall. Learning distance functions for image retrieval. In *CVPR*, 2004. 1786

[16] B. Kulis, P. Jain, and K. Grauman. Fast similarity search for learned metrics. *IEEE PAMI*, 39(12):2143–2157, 2009. 1786, 1787

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004. 1790

[18] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009. 1786

[19] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *Proc. ECCV*, 2010. 1785, 1786, 1787, 1788, 1789, 1790, 1791

[20] M. Stark, M. Goesele, and B. Schiele. A shape-based object class model for knowledge transfer. In *ICCV*, 2009. 1786

[21] J. Yang, R. Yan, and A. G. Hauptmann. Cross-domain video concept detection using adaptive svms. *ACM Multimedia*, 2007. 1785, 1786, 1787

A. Appendix: Proofs

Proof of Lemma 3.1: Let W have singular value decomposition $U\Sigma\tilde{U}^T$. We can therefore write W as $W = \sum_{j=1}^{p} \sigma_j \boldsymbol{u}_j \tilde{\boldsymbol{u}}_j^T$, where p is the rank of W. Every $\boldsymbol{u}_j$ is either in the range space of X or the null space of X. If it is in the range space, then $\boldsymbol{u}_j = X\boldsymbol{z}_j$, for some $\boldsymbol{z}_j$; if it is in the null space, then $X^T\boldsymbol{u}_j = 0$. An analogous statement holds for $\tilde{\boldsymbol{u}}_j$ and Y.

Consider $c_i(X^TWY)$, and expand W via the SVD:

$$X^TWY = X^T\left(\sum_{j=1}^{p} \sigma_j \boldsymbol{u}_j \tilde{\boldsymbol{u}}_j^T\right)Y = \sum_{j=1}^{p} \sigma_j (X^T \boldsymbol{u}_j \tilde{\boldsymbol{u}}_j^T Y).$$

If either $\boldsymbol{u}_j$ is in the null space of X or $\tilde{\boldsymbol{u}}_j$ is in the null space of Y, then the corresponding term in the sum will be zero. As a result, σ_j has no impact on the value of $c_i(X^TWY)$, and only impacts the $r_j(\sigma_j)$ term of the objective. Since the minimizer of r_j is at 0, we set $\sigma_j = 0$ in this case.

Therefore, let us assume that the singular values are ordered so that the first t are such that the corresponding singular vectors $\boldsymbol{u}$ are in the range space of X and $\tilde{\boldsymbol{u}}$ are in the range space of Y. The remainder of the singular values are set to 0 by the above argument. Then we have

$$\begin{aligned} W &= \sum_{j=1}^{t} \sigma_j \boldsymbol{u}_j \tilde{\boldsymbol{u}}_j^T = \sum_{j=1}^{t} \sigma_j X \boldsymbol{z}_j \tilde{\boldsymbol{z}}_j^T Y^T \\ &= X\left(\sum_{j=1}^{t} \sigma_j \boldsymbol{z}_j \tilde{\boldsymbol{z}}_j^T\right) Y^T = X\tilde{L}Y^T, \end{aligned}$$

where $\tilde{L} = \sum_{j=1}^{t} \sigma_j \boldsymbol{z}_j \tilde{\boldsymbol{z}}_j^T$. With the transformation $L = K_A^{1/2}\tilde{L}K_B^{1/2}$, we can equivalently write as $W = XK_A^{-1/2}LK_B^{-1/2}Y^T$ (this transformation will simplify the theorem proof), proving the lemma.

Proof of Theorem 3.2: Denote $V_A = XK_A^{-1/2}$ and $V_B = YK_B^{-1/2}$. Note that V_A and V_B are orthogonal matrices. From the lemma, $W = V_ALV_B^T$; let $V_A^\perp$ and $V_B^\perp$ be the orthogonal complements to V_A and V_B, and let $\bar{V_A} = [V_A \;\; V_A^\perp]$ and $\bar{V_B} = [V_B \;\; V_B^\perp]$. Then

$$r\left(\bar{V_A}\begin{bmatrix} L & 0 \\ 0 & 0 \end{bmatrix}\bar{V_B}^T\right) = r\left(\begin{bmatrix} W & 0 \\ 0 & 0 \end{bmatrix}\right) = r(W) + const.$$

One can easily verify that, given two orthogonal matrices V_1 and V_2 and an arbitrary matrix M, that $r(V_1MV_2) = \sum_j r_j(\sigma_j)$ if σ_j are the singular values of M. So

$$r\left(\bar{V_A}\begin{bmatrix} L & 0 \\ 0 & 0 \end{bmatrix}\bar{V_B}^T\right) = \sum_j r_j(\bar{\sigma_j}) + const = r(L) + const,$$

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

Finally, rewrite the constraints c_i using $W = XK_A^{-1/2}LK_B^{-1/2}Y^T$:

$$c_i(X^TWY) = c_i(K_AK_A^{-1/2}LK_B^{-1/2}K_B) = c_i(K_A^{1/2}LK_B^{1/2}).$$

The theorem follows by rewriting r and the c_i functions using the above derivations in terms of L. Note that both r and the c_i's can be computed independently of the dimensionality, so simple arguments show that the optimization may be solved in polynomial time independent of the dimensionality when the r_j functions are convex.