Discovering Latent Domains for Multisource Domain Adaptation

Judy Hoffman[1,2], Brian Kulis[4], Trevor Darrell[1,2], and Kate Saenko[1,2,3]

UC Berkeley EECS[1], ICSI[2], Berkeley, California
Harvard University[3], Cambridge, Massachusetts
Ohio State University[4], Columbus, Ohio
`jhoffman,saenko,trevor@eecs.berkeley.edu`
`kulis@cse.ohio-state.edu`

Abstract. Recent domain adaptation methods successfully learn cross-domain transforms to map points between source and target domains. Yet, these methods are either restricted to a single training domain, or assume that the separation into source domains is known a priori. However, most available training data contains multiple unknown domains. In this paper, we present both a novel domain transform mixture model which outperforms a single transform model when multiple domains are present, and a novel constrained clustering method that successfully discovers latent domains. Our discovery method is based on a novel hierarchical clustering technique that uses available object category information to constrain the set of feasible domain separations. To illustrate the effectiveness of our approach we present experiments on two commonly available image datasets with and without known domain labels: in both cases our method outperforms baseline techniques which use no domain adaptation or domain adaptation methods that presume a single underlying domain shift.

1 Introduction

Despite efforts to the contrary, most image datasets exhibit a clear *dataset bias*: supervised learning on a particular dataset nearly always leads to a significant loss in accuracy when the models are tested in a new domain [1, 2]. *Domain Adaptation* methods have been proposed as a solution to dataset bias and are becoming increasingly popular in computer vision. Especially attractive are recent weakly-supervised methods that learn to transform features between domains based on partially labeled data [1, 3, 4].

A major limitation of these methods is the assumption that the domain/dataset label is provided for each training image. However, in practice, one often has access to large amounts of object labeled data composed of multiple *unknown domains.* For example, images found on the web can be thought of as a collection of many hidden domains. As shown in Figure 1, image search results for "person" and "bicycle" consist of several *types*, such as close-up photos of a face, photos of an entire human figure, group shots, line drawings of a person, etc.

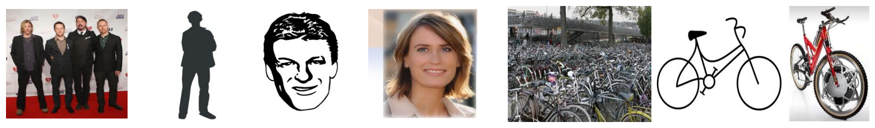


Fig. 1. Training images for object recognition may contain several unknown domains, such as the line drawings, close-up photos and far-away shots returned by web image search for *person* and *bicycle*.

Existing methods have not addressed the problem of separating such data into latent domains for the purpose of domain adaptation.

Even when the separation into source domains is known, the above methods are limited to a single adaptive transform between source and target data, computed by either simply pooling [1, 3] or averaging [4] multiple sources into one source domain. Learning a single transform may be sub-optimal in approximating multiple domain shifts. Consider a simple example: two source image datasets, one with *low-resolution*, *color* images and one with *high-resolution*, *grayscale* images. When mapping images from a novel *high-resolution*, *color* domain to each of these source domains, we would like a transform that either discounts the high-resolution details or the color, but not both. Rather than force a single transform to perform both mappings, a richer model with multiple separate transforms is desirable.

The contribution of this paper is two-fold: First, we propose a method that discovers latent domain labels in data which has heterogeneous but unknown domain structure. We use a probabilistic framework to derive a hierarchical constrained assignment algorithm that separates heterogeneous training data into latent clusters, or domains. Our second contribution is an extension of the feature-transform method in [3] to multi-domain adaptation that uses source domain labels to define a mixture-transform model. Tests on standard datasets with known domain labels confirm that our method is more accurate at discovering domain structure than baseline clustering methods, and that our transform mixture model outperforms a single transform approach. When domain labels are unknown, we evaluate the end-to-end recognition performance with no adaptation baseline, baseline adaptation, and transform-mixture adaptation with discovered domain labels (our method). Our experiments on category data from the *Bing* image search dataset confirm improved classification performance using the inferred domains.

2 Related Work

Domain adaptation aims to compensate for covariate shift, i.e. a change in the feature distribution from training to test time, which is a version of the more general dataset shift problem [5].

Several methods have been developed in the NLP community, e.g., structural correspondence learning was proposed for NLP tasks such as sentiment classification [6]. Daume [7] introduced a feature replication method for domain adaptation, which has been applied to both NLP and vision tasks. The basic idea is: given a feature vector $\boldsymbol{x}$, define the augmented feature vector $\tilde{\boldsymbol{x}} = (\boldsymbol{x}; \boldsymbol{x}; \mathbf{0})$ for data points in the source and $\tilde{\boldsymbol{x}} = (\boldsymbol{x}; \mathbf{0}; \boldsymbol{x})$ for data points in the target.

Recently, several authors have developed approaches for vision tasks, and the field is becoming increasingly aware of existing dataset bias [8]. These methods can be roughly categorized as classifier adaptation approaches and feature-transform approaches. The former approach adapts the parameters of each per-category binary classifier, typically a support vector machine, and includes methods like a weighted combination of source and target SVMs; transductive SVMs applied to adaptation in [9]; Adaptive SVM [10], where the parameters are adapted by adding a perturbation function; Domain Transfer SVM [11], which learns a target decision function while reducing the mismatch in the domain distributions; and a related method [12] which utilizes adaptive multiple kernel learning to learn a kernel function based on multiple base kernels. Classifier adaptation approaches are typically supervised, requiring labeled examples for each category in both source and target.

On the other hand, feature transform approaches map or translate the input features directly between domains, and then apply a classifier [1,3,4,13,14]. The advantage of these approaches over the classifier-based ones above is that they are able to transfer the adaptive transformation to novel problems, and even deal with new feature types and dimensionalities. For example, the asymmetric nonlinear transform method [3] learns a domain-independent similarity function between arbitrary feature sets based on class constraints. In this paper we focus on transform learning, as it can be scaled up to many novel object categories and heterogeneous features. Other work on feature-translation has included translating features between camera views to transfer activity models [13], and translating user preferences between text and image domains [14].

Some of the existing methods can accommodate multiple source domains, if they are known. The feature replication method [7] can be easily extended to multiple sources. In methods based on combinations of SVM classifiers, this is done primarily by weighting the classifiers learned on each source domain, either equally, or as in [15], using the maximum mean discrepancy (MMD) criterion, which measures the distance between distributions. The A-MKL [12] learns an optimal weighted combination of the pre-learned classifiers, however, in the evaluation the classifiers came from the same source domain. The unsupervised approach based on Grassman manifolds presented by Gopalan et al. [4] supports multiple sources by first computing the Karcher mean of the sources and then running the single source version of their algorithm.

None of the existing methods address the case of domain adaptation with unknown domain labels, the main focus of our paper. We also present the first multi-domain version of the asymmetric nonlinear transform [3], which so far has been limited to the single domain case.

Our domain discovery method is based on a constrained clustering method, a topic of active research for several years. The most related work to our approach is based on constrained k-means. The method of Wagstaff et al. [16] is the earliest constrained clustering method based on k-means; their algorithm greedily assigns points to the closest cluster which does not violate any constraints. Improvements to this basic algorithm were considered in [17, 18]. A variant of constrained k-means was also used recently with soft constraints to learn more discriminative codebooks [19]. In general, the constraints in these approaches are incorporated as additional penalty terms to the k-means objective function (and therefore are not guaranteed to be satisfied), and algorithms are developed to minimize the penalized k-means objective. In contrast to existing work on constrained k-means, our algorithm is guaranteed to find a clustering that satisfies the constraints and provably converges locally under certain assumptions. An overview of recent semi-supervised clustering techniques, including techniques not based on k-means, can be found in [20].

2.1 Single Transform Domain Adaptation

We review the single-transform method of Kulis et al. [3]. Suppose we have a source domain $\mathcal{A}$ containing observations $\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_\mathcal{A}}$. Similarly, let $\mathcal{B}$ be the target domain, containing observations $\boldsymbol{y}_1, ..., \boldsymbol{y}_{n_\mathcal{B}}$. Suppose we are also given a set of labels $l_1^\mathcal{A}, ..., l_{m_\mathcal{A}}^\mathcal{A}$ for the source observations and a partial set of labels $l_1^\mathcal{B}, ..., l_{m_\mathcal{B}}^\mathcal{B}$ for the target observations, with $m_\mathcal{B} < m_\mathcal{A}$ and $l \in \{1, ..., K\}$, where K is the number of categories. The goal of domain transform learning is to estimate a semantic similarity function $sim(\boldsymbol{x}, \boldsymbol{y})$ which outputs high similarity values for pairs of source examples $\boldsymbol{x}$ and novel target examples $\boldsymbol{y}$ if they have the same label, and low similarity values if they have different labels.

Following [3], consider a similarity function between source and target data parametrized by a matrix W, i.e., $sim_W(\boldsymbol{x}, \boldsymbol{y}) = \phi_\mathcal{A}(\boldsymbol{x})^T W \phi_\mathcal{B}(\boldsymbol{y})$, where $\phi_\mathcal{A}$ and $\phi_\mathcal{B}$ map examples from the source and target, respectively, into kernel space. In general, mapping to kernel space makes it possible to learn non-linear similarity mappings in the input space.

The transform W can be learned via a regularized optimization problem with loss functions that depend on sim_W; that is, by optimizing for a matrix W which minimizes $r(W) + \gamma \sum_{i=1}^{C} c_i(\phi_\mathcal{A}(X)^T W \phi_\mathcal{B}(Y))$, where r is a matrix regularizer, C is the number of constraints, $\phi_\mathcal{A}(X)$ is the matrix of source data mapped to kernel space, $\phi_\mathcal{B}(Y)$ is the matrix of target data mapped to kernel space, and c_i are loss functions over the matrix $\phi_\mathcal{A}(X)^T W \phi_\mathcal{B}(Y)$. Though the model can incorporate various constraints/losses, it is customary to focus on constraints that penalize small values of $sim_W(\boldsymbol{x}, \boldsymbol{y})$ when $\boldsymbol{x}$ and $\boldsymbol{y}$ share a class label, and penalize large values of $sim_W(\boldsymbol{x}, \boldsymbol{y})$ when they have different labels. For a particular class of regularizers r, this problem may be efficiently optimized in kernel space; see [3].

3 Multiple Transform Domain Adaptation

We first extend the single source feature transform method [3] to a multi-source algorithm, considering the case of known domain labels, and then in Section 4 we present our main contribution: a method for discovering unknown domain labels.

Transform-based domain adaptation captures category-independent domain shift information, which can be generalized to a held-out category for which no labels are available in a new domain. The single-transform method described in Section 2.1 is sub-optimal when multiple domain shifts are present. We remedy this by constructing a domain transform mixture model as follows.

Suppose we have a set of domains $\mathcal{A}_1, ..., \mathcal{A}_S$ containing observations $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$ and labels $\{\ell_1^{\mathcal{A}}, ..., \ell_n^{\mathcal{A}}\}$. Let $\boldsymbol{a} = \{a_1, ..., a_n\}$ specify the domain of each observation, such that $a_i \in \{1, ..., S\}$.

We start by using the single source feature transform method to learn a transformation W_k for each source $k \in \{1, \dots, S\}$. Each of the S different object category classifiers outputs a multi-class probability for a given test point, $\boldsymbol{y}$. We denote the output probability over classes, c, of the classifier learned for the kth domain as, $p(c|d = k, \boldsymbol{y})$.

We classify a novel test point from a target domain by considering the domain as a latent variable that is marginalized out as follows:

$$\text{label}(\boldsymbol{y}) = arg \max_{c \in \{1:K\}} p(c|\boldsymbol{y}) \tag{1}$$

$$= arg \max_{c \in \{1:K\}} \sum_{k=1}^{S} p(d = k|\boldsymbol{y}) p(c|d = k, \boldsymbol{y}) \tag{2}$$

Where $p(d|\boldsymbol{y})$ is the output of a domain label classifier and $p(c|d, \boldsymbol{y})$ is the output of a domain specific object model based on the learned transforms. For example, if the domain specific classifier is (kernelized) nearest neighbors, as in our experiments, and we let $\boldsymbol{x}_k^*(\boldsymbol{y})$ be the most similar point in domain k to test point $\boldsymbol{y}$, then our domain specific object category probability can be expressed as:

$$p(c|d = k, \boldsymbol{y}) = \frac{\phi_A(\boldsymbol{x}_k^*(\boldsymbol{y}))^T W_j \phi_B(\boldsymbol{y})}{\sum_{k'=1}^{S} \phi_A(\boldsymbol{x}_{k'}^*(\boldsymbol{y}))^T W_j \phi_B(\boldsymbol{y})} \tag{3}$$

Finally, to obtain $p(d = k|\boldsymbol{y})$ we train an SVM classifier with probabilistic outputs, using known domain labels and source training data. In summary, category classification of a test point $\boldsymbol{y}$ amounts to a weighted sum of the probability of a particular category given that the point is from a particular domain, where the weights are the probability that the test point belongs to each domain.

4 Domain Clustering

For datasets which do not have domain labels, we must infer domain assignments $\hat{\boldsymbol{a}}$ that best approximate the true assignments, $\boldsymbol{a}$. This task is difficult because,

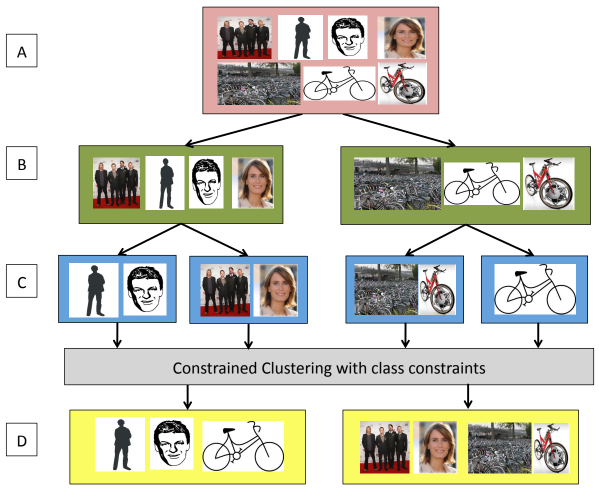


Fig. 2. An illustration of our approach for discovering latent domains: Given input images with unknown domains types (A), we separate examples according to their semantic category labels (B). Then, we cluster the data from each semantic category group to produce local clusters (C), and, finally, use do-not-link constraints between clusters of same category to produce the domain specific clusters (D). This example shows separation of *person* and *bicycle* search results into two domains of line drawing and natural images. Our algorithm iterates between steps (B-D) until convergence.

in many cases, the data is naturally separated according to semantic categories. That is, a standard clustering method such as k-means or EM for mixtures of Gaussians would tend to return clusters based on the semantic category labels, which is clearly undesirable.

High-level Description: We propose a two-stage approach for domain discovery (See Figure 2 for illustration of algorithm). The idea is as follows: if there are K semantic categories and S domains in the data, we will group the data points $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$ into $J = K \cdot S$ *local clusters*—intuitively, each local cluster will contain data points from one domain and one semantic category (step B-C in Figure 2). In the second stage of clustering (*domain clustering*), we will cluster the means of the local clusters, in order to find domains (step C-D in Figure 2). The two stages are iterated to convergence. To make this approach consistent with our goals, we add two crucial constraints to the clustering procedure: a) we require that each local cluster only contains data points from a single category, and b) we constrain each domain cluster to contain only one local cluster from each object category.

Formal Definition: Let us now specify this model more formally. Define a local latent variable $z^L_{ij} \in \{0, 1\}$ to indicate whether observation $\boldsymbol{x}_i$ should be assigned to local cluster j. Let the domain latent variable $z^G_{jk} \in \{0, 1\}$ indicate

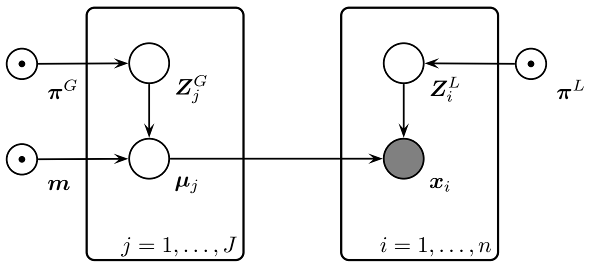


Fig. 3. Graphical model for domain discovery. The data points x_i are generated from category specific local mixtures, whose means are assumed to be drawn from a latent global (*domain*) mixture.

whether local cluster j is assigned to domain k. Further, let $\boldsymbol{\mu}_j$ be the mean of the observations in local cluster j and $\boldsymbol{m}_k$ be the mean of domain cluster k. Finally, let the mixing weights for the global clusters be given by π_k^G, for $k = 1, ..., S$, and let the mixing weights for the local clusters be given by π_j^L for $j = 1, ..., J$. See Figure 3 for a graphical depiction of our model. Note that, unlike a standard Gaussian mixture model, each cluster mean $\boldsymbol{\mu}_j$ is itself a data point within a mixture model. For simplicity, we are assuming that all clusters have a global covariance of σI—one could easily extend the model to learn the covariances. The conditional probabilities in our graphical model are defined as:

$$p(\boldsymbol{z}_i^L \mid \boldsymbol{\pi}^L) = \prod_{j=1}^{J} (\pi_j^L)^{z_{ij}^L} \qquad p(\boldsymbol{z}_j^G \mid \boldsymbol{\pi}^G) = \prod_{k=1}^{S} (\pi_k^G)^{z_{jk}^G} \tag{4}$$

$$p(\boldsymbol{x}_i \mid \boldsymbol{z}_i^L, \boldsymbol{\mu}) = \prod_{j=1}^{J} \mathcal{N}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \sigma I)^{z_{ij}^L} \tag{5}$$

$$p(\boldsymbol{\mu}_j \mid \boldsymbol{z}_j^G, \boldsymbol{m}) = \prod_{k=1}^{S} \mathcal{N}(\boldsymbol{\mu}_j \mid \boldsymbol{m}_k, \sigma I)^{z_{jk}^G} \tag{6}$$

We may think of this model in a generative manner as follows: for each local cluster $j = 1, ..., J = S \cdot K$, we determine its underlying domain via the $\boldsymbol{\pi}^G$ mixing weights. Given this domain, we generate $\boldsymbol{\mu}_j$, the mean for the local cluster. Then, to generate the data points $\boldsymbol{x}_i$, we first determine which local cluster j the data point belongs to using $\boldsymbol{\pi}^L$, and then we generate the point from the corresponding $\boldsymbol{\mu}_j$.

At this point, we still need to add the constraints discussed above, namely that the local clusters only contain data points from a single category and that domain clusters contain only a single local cluster from each category. Such constraints can be difficult to enforce in a probabilistic model, but are considerably simpler in a hard clustering model. For instance, in semi-supervised clustering there is ample literature on adding constraints to the k-means objective via constraints or penalties [16, 18]. Thus, we will utilize a standard asymptotic argument on our probabilistic model to create a corresponding hard clustering model. In particular, if one takes a mixture of Gaussian model with fixed σI covariances across clusters, and lets $\sigma \to 0$, the expected log likelihood approaches the k-means objective, and the EM algorithm approaches the k-means algorithm. In an analogous manner, we will consider the log-likelihood of our model. Let $\boldsymbol{Z}^L = \{\boldsymbol{z}_i^L\}$, $\boldsymbol{Z}^G = \{\boldsymbol{z}_j^G\}$, and $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n]$.

$$\ln\left[p(\boldsymbol{Z}^G, \boldsymbol{Z}^L, X \mid \boldsymbol{m}, \boldsymbol{\mu})\right] = \tag{7}$$
$$\sum_{i=1}^{n}\sum_{j=1}^{J}\sum_{k=1}^{S}\left(z_{jk}^G(\ln \pi_k^G + \ln \mathcal{N}(\boldsymbol{\mu}_j \mid \boldsymbol{m}_k, \sigma I)) + z_{ij}^L(\ln \pi_j^L + \ln \mathcal{N}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \sigma I))\right)$$

If we have no prior knowledge about the domains, then we assume the mixing weights π^G and π^L to be uniform. Then to create a hard clustering problem we let σ tend to 0 and taking expectations with respect to $\boldsymbol{Z}^G$ and $\boldsymbol{Z}^L$, we obtain the following hard clustering objective:

$$\min_{\boldsymbol{Z}^G, \boldsymbol{Z}^L, \boldsymbol{\mu}, \boldsymbol{m}} \sum_{i=1}^{n}\sum_{j=1}^{J} \boldsymbol{Z}_{ij}^L (x_i - \mu_j)^2 + \sum_{j=1}^{J}\sum_{k=1}^{S} \boldsymbol{Z}_{jk}^G (\mu_j - \boldsymbol{m}_k)^2 \tag{8}$$
$$\text{subject to: } \forall j,k: \ \boldsymbol{Z}_{jk}^G \in \{0,1\}, \quad \forall i,j: \ \boldsymbol{Z}_{ij}^L \in \{0,1\} \tag{9}$$
$$\forall j: \ \sum_{k=1}^{S} \boldsymbol{Z}_{jk}^G = 1, \quad \forall i: \ \sum_{j=1}^{J} \boldsymbol{Z}_{ij}^L = 1 \tag{10}$$
$$\tag{11}$$

The constraints above are standard hard assignment constraints, saying that every observation must be assigned to one local cluster and every local cluster must be assigned to only one global cluster. Now that we have transformed the clustering problem into a hard clustering model, we can easily add constraints on the $\boldsymbol{Z}^G$ and $\boldsymbol{Z}^L$ assignments, leading to our final model.

$$\min_{\boldsymbol{Z}^G, \boldsymbol{Z}^L, \boldsymbol{\mu}, \boldsymbol{m}} \sum_{i=1}^{n}\sum_{j=1}^{J} \boldsymbol{Z}_{ij}^L (x_i - \mu_j)^2 + \sum_{j=1}^{J}\sum_{k=1}^{S} \boldsymbol{Z}_{jk}^G (\mu_j - \boldsymbol{m}_k)^2 \tag{12}$$
$$\text{subject to: } \forall j,k: \ \boldsymbol{Z}_{jk}^G \in \{0,1\}, \quad \forall i,j: \ \boldsymbol{Z}_{ij}^L \in \{0,1\} \tag{13}$$
$$\forall j: \ \sum_{k=1}^{S} \boldsymbol{Z}_{jk}^G = 1, \quad \forall i: \ \sum_{j=1}^{J} \boldsymbol{Z}_{ij}^L = 1 \tag{14}$$
$$\forall j: \ \sum_{i=1}^{n} \delta(l_i \neq l_j) \boldsymbol{Z}_{ij}^L = 0 \tag{15}$$
$$\forall k: \ \sum_{c=1}^{K}\sum_{j=1}^{J} \delta(\text{label}(j) = c) \boldsymbol{Z}_{jk}^G = 1 \tag{16}$$

Equation (15) says that all observations assigned to a single local cluster j must contain the same object category label. Finally, we add the constraint in Equation (16), which says that each global cluster should only contain one local cluster from each object category. Here, we define label(j) to mean the object category label of the observations assigned to the local cluster j. Together the last two constraints restrict the feasible global clustering solutions to be the solutions that contain observations from every object category while placing observations together that are close according to a Euclidean distance.

To solve this optimization problem we formulate an EM style iteration algorithm. We first initialize $\boldsymbol{Z}^G, \mu, \boldsymbol{m}$ and then perform the following updates until convergence (or until the objective doesn't change more than some threshold between iterations):

$\boldsymbol{Z}^L_{ij}$: For each observation $\boldsymbol{x}_i$, set $\boldsymbol{Z}^L_{ij} = 1$ for the particular j that minimizes $(\boldsymbol{x}_i - \mu_j)^2$.

μ_j: For each local cluster j, set $\mu_j = \frac{\sum_i \boldsymbol{Z}^L_{ij}\boldsymbol{x}_i + \sum_k \boldsymbol{Z}^G_{jk}\boldsymbol{m}_k}{\sum_i \boldsymbol{Z}^L_{ij} + \sum_k \boldsymbol{Z}^G_{jk}}$

$\boldsymbol{Z}^G_{jk}$: For each local cluster j, set $\boldsymbol{Z}^G_{jk} = 1$ for the particular global cluster that minimizes $\boldsymbol{Z}^G_{jk}(\mu_j - \boldsymbol{m}_k)^2$ while satisfying Equation (16). For the case where the number of latent domains is assumed small, we can try all possible assignments of local to global clusters at this stage.

$\boldsymbol{m}_k$: For each global cluster k, set $\boldsymbol{m}_k = \frac{\sum_j \boldsymbol{Z}^G_{jk}\mu_j}{\sum_j \boldsymbol{Z}^G_{jk}}$

Note, these updates correspond to constrained versions of the EM updates from the probabilistic model as $\sigma \to 0$. In general adding constraints to EM updates can cause the objective to diverge. However, for our problem, when the number of domains, S, is small (which is a practical assumption for many domain adaptation tasks) these updates can be shown to provably converge locally (see supplemental material). In Section 5 we show experimentally that even for large heterogeneous data sources learning to separate into a small number of latent domains is sufficient in practice.

We evaluate absolute domain discovery performance compared against a constrained hard HDP method [21] and a standard unconstrained k-means. The hard HDP seeks to minimize the distance between data points and their local mean plus data points to their assigned global mean, whereas we introduce an intermediate latent variable for the local cluster means. That, combined with the fact that constrained hard HDP can create new global clusters, causes the constrained hard HDP to degenerate to the case where each global cluster contains only one local cluster. In Section 5 we present clustering accuracy experiments that show our method does in fact provide significant benefit over the hard HDP method and a standard k-means baseline.

5 Experiments

We present the following three experiments. First, we evaluate our multidomain mixture model extension of [3] described in Section 3 for the case where the domain labels are known. We compare this to using the single-transform method in

[3]. Next, we evaluate the ability of our constrained clustering method to recover domains by mixing datasets and omitting the dataset labels, and compare to two baseline clustering methods. Finally, we evaluate the end-to-end recognition performance on a dataset with unknown domains, compared against methods with no adaptation, baseline adaptation, and our multiple transform adaptation with discovered domain labels.

Datasets: For our experiments we used two different data sets. First we used the *Office* data set created by [1] specifically for object domain adaptation, which we modified to artificially introduce more domains. We use the code provided by the authors to extract features and to implement the single-domain asymmetric transform method. Second, we evaluate on a subset of the *Bing-Caltech256* data set created by [9], using the authors' original image features. The *Bing* dataset is comprised of search results from object label keywords and therefore has many sub-domains without any explicit labels.

The *Office* dataset contains three domains with 31 object categories: Amazon (a), which contains images from amazon.com; Digital SLR (d), and Webcam (w). Webcam and Digital SLR contain images of the same 5 instances for each object category, but vary in pose, lighting, and camera. In addition to these three domains, we also created two more domains which are artificially modified versions of two the the original domains: Webcam-blur (Wb), which contains the blurred version of all webcam data images (using a Gaussian filter of width 5), and Amazon-rotflip (Ar), which contains the result of rotating all amazon images by 10-20 degrees and then creating their mirror image.

Object Classification using *Office*: For our first object classification experiment, we follow the experimental procedure for semi-supervised learning described in [1, 4]: we train the transforms on categories 1-15, with 10 instances per category in each source domain and 5 instances per category in the target domain, and test them using the classifier described in Section 3 on the rest of the categories.

The experiment measures the accuracy of predicting the labels of target data points with no examples of the correct label within the target domain, representing the transform's ability to capture general shifts rather than simply the source shift's effect on a particular object category.

We compared our method to the single-transform method of [3] and the semi-supervised multi-domain version of [4]. The γ parameter of ours and [3] was optimized for the single transform baseline in all experiments. We ran code provided by the authors of [4]; we experimented with over 1000 combinations of the four parameter settings within the ranges specified in the code from the author's website and the paper, but were unable to produce resulting accuracies above the standard KNN with no adaptation baseline for any of the domain shifts presented, and so we omit these results from the following figure.

Figure 4 shows the absolute improvement in accuracy of the supervised multiple-domain method and a single transform method over the KNN classifier using Euclidean distance (i.e., no adaptation) for various domain shifts, averaged over 10 train/test splits for each shift. Overall, the average accuracy

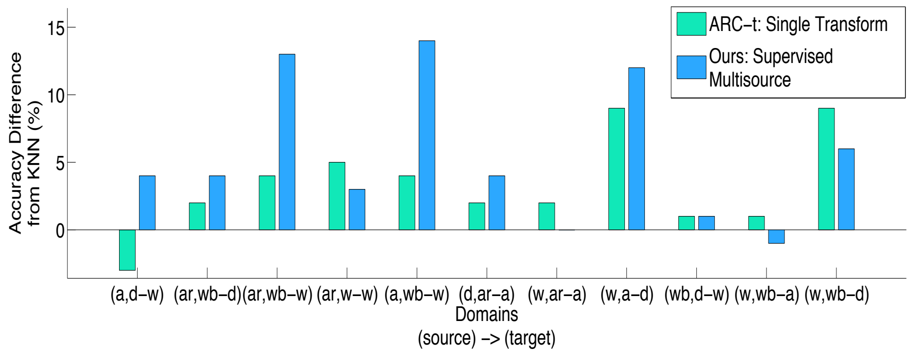


Fig. 4. Classification Accuracy experiment on *Office* dataset with known domain labels. The plot shows the average accuracy improvement of each method over a baseline KNN. Each cluster of bars is the results for a different domain shift, where the domains are indicated on the x-axis. For each of these cases the parameter, γ was set to be the value that maximized the single transform method baseline. The average accuracy gained over KNN across all datasets was 3.27 for the single source baseline [3] and 5.45 for our multisource domain-supervised method. For some datasets our method outperforms [3] significantly, achieving up to a 18.9% relative improvement over KNN.

gained over KNN was 3.27 for the single source baseline [3] and 5.45 for our domain-supervised method. The average KNN classification accuracy was 28.9, which means these absolute improvements correspond to an 11.3% and 18.9% relative improvement over KNN for each method respectively.

We notice that shifts where the supervised method achieves the most gains (up to 48% relative gain in accuracy) have higher separability of the source domains via clustering reported below (in Figure 5.)

Clustering: We begin with an analysis of our clustering algorithm on all pairs of the 5 domains in the *Office* dataset, for which we know the ground truth source labels. Figure 5 plots the accuracy improvement over chance between the ground truth domain labels and the cluster labels learned using our constrained clustering algorithm. The results are averaged over 10 runs with different data splits for each run. As baselines, we implemented standard, unconstrained, k-means, as well as, a hard HDP [21] with the same do-not-link class based constraints presented in Section 4.

Our clustering method outperforms these baselines and in general has high accuracy when compared with the ground truth domain labels. The one exception is when clustering the pair (Amazon-rotflip, Amazon). In this case, the domains represented in our feature space are very similar and so there is no adequate separation for the clustering algorithm to find. The last bar in Figure 5 shows the mean clustering accuracy across all domain shifts and shows the significant improvement of using our method to cluster into domains.

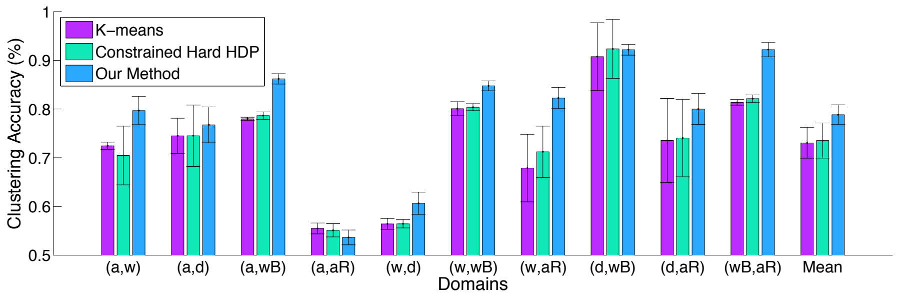


Fig. 5. Clustering quality, measured in terms of accuracy of cluster labels. Each cluster of bars represents a mixed pair of domains from the *Office* dataset, see text. The results are averaged over multiple data splits. Our method significantly outperforms the baselines in 7 out of the 10 datasets.

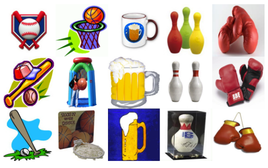
Object Classification using *Bing-Caltech256*: We show the results of running our end-to-end algorithm on the Bing web search and Caltech256 dataset created by [9]. Here we assume the Bing images contain sub-domains. We use the first 30 categories from each of Bing and Caltech, and assume that all points in Bing have strict class labels (in reality, the results of web search are only weakly labeled.) There are no domain labels for the *Bing* source dataset so we present qualitative clustering results in Figures 6(a-c). The clusters shown represent intuitive domain distinctions, justifying out method: 6(a) cartoon/drawing/synthetic images with limited illumination, 6(b) product images with simple backgrounds and some illumination, 6(c) natural or cluttered scenes.

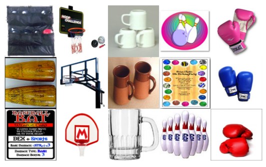
After clustering, we train the transforms on the first 15 categories in the Caltech256 domain and use categories 16 to 30 for testing, again testing the prediction accuracy of transformed data points into new object categories. We set $S = 2$ and use the SVM-based domain classifier described in Section 3 for the mixture model.

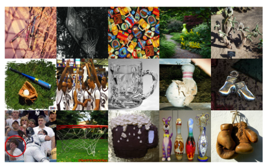
Figure 6 plots the results of running our algorithm with 20 instances per category in the source domain and 10 instances per category in the target domain. Ideally we would cross-validate the learning parameter γ, but due to lack of time we report results over varying γ on the test set. Our method obtains better results for a larger range of γ. The best performance obtained by the baseline is around 40% accuracy, while our method obtains 44% accuracy. This shows the advantage of using our method instead of pooling the data to learn a single transform, and that it is possible to learn domain clusters and to use them to effectively produce a better-adapted object classifier.

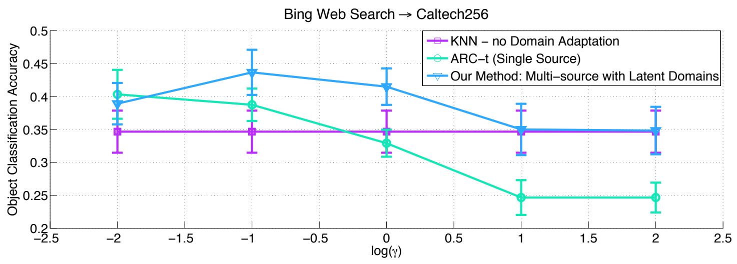
6 Conclusion

In this paper we presented a novel constrained clustering algorithm to distinguish unique domains from a large heterogeneous domain. Additionally, we proposed a


(a) Cluster 1


(b) Cluster 2


(c) Cluster 3



(d) Classification Result

Fig. 6. Results on the *Bing-Caltech256* dataset. (a-c) show example images from clustering the *Bing* dataset into three clusters. For each domain cluster the the three images closest to the domain cluster mean are shown from the categories baseball bat, basketball hoop, beer mug, bowling pin, and boxing glove are shown. The clusters represent intuitive domain definitions: (a) cartoon/drawings/synthetic images with limited illumination, (b) product images with simple backgrounds and some illumination, (c) natural or cluttered scenes. (d) Shows average accuracy versus the log of the learning parameter γ. Our unsupervised multi-domain method outperforms the use of a single learned transform, which falls below the KNN baseline as more weight is put on the constraints in the optimization problem.

simple multisource extension to the nonlinear transform-based ARC-t method of [1, 3]. Our algorithm provides the ability to separate heterogeneous source data and learn multiple transforms, which creates a more accurate mapping of the source data onto the target domain than a single transform map. Our experiments illustrate that the multiple transformations can be effectively applied to new object categories at test time.

Thus far our experiments have focused on the case of multiple domains in the source and only one domain in the target. In future work, we plan to test our algorithm with multiple target domains. This should be a direct extension by clustering the target domain using the method described in Section 4.

References

1. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: Proc. ECCV. (2010)
2. Torralba, A., Efros, A.: Unbiased look at dataset bias. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2011)

3. Kulis, B., Saenko, K., Darrell, T.: What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In: Proc. CVPR. (2011)
4. Gopalan, R., Li, R., Chellappa, R.: Domain adaptation for object recognition: An unsupervised approach. In: 13th International Conference on Computer Vision 2011. (2011)
5. Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D.: Dataset Shift in Machine Learning. The MIT Press (2009)
6. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. ACL (2007)
7. Daume III, H.: Frustratingly easy domain adaptation. In: ACL. (2007)
8. Torralba, A., Efros, A.: Unbiased look at dataset bias. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2011)
9. Bergamo, A., Torresani, L.: Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In: Neural Information Processing Systems (NIPS). (2010)
10. Yang, J., Yan, R., Hauptmann, A.G.: Cross-domain video concept detection using Adaptive SVMs. ACM Multimedia (2007)
11. Duan, L., Tsang, I.W., Xu, D., Maybank, S.J.: Domain transfer SVM for video concept detection. In: CVPR. (2009)
12. Duan, L., Xu, D., Tsang, I., Luo, J.: Visual event recognition in videos by learning from web data. In: Proc. CVPR. (2010)
13. Farhadi, A., Tabrizi, M.K.: Learning to recognize activities from the wrong view point. In: ECCV. (2008)
14. Dai, W., Chen, Y., Xue, G.R., Yang, Q., Yu, Y.: Translated learning: Transfer learning across different feature spaces. In: Proc. NIPS. (2008)
15. Duan, L., Tsang, I.W., Xu, D., Chua, T.S.: Domain adaptation from multiple sources via auxiliary classifiers. In: Proceedings of the 26th Annual International Conference on Machine Learning. ICML '09 (2009)
16. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained k-means clustering with background knowledge. In: Proc. ICML. (2001)
17. Basu, S., Bilenko, M., Mooney, R.: A probabilistic framework for semi-supervised clustering. In: Proc. 22nd SIGKDD Conference. (2004)
18. Kulis, B., Basu, S., Dhillon, I., Mooney, R.: Semi-supervised graph clustering: A kernel approach. In: Proc. ICML. (2005)
19. Taralova, E., De la Torre Frade, F., Hebert, M.: Source constrained clustering. In: 13th International Conference on Computer Vision 2011. (2011)
20. Basu, S., Davidson, I., Wagstaff, K., eds.: Constrained Clustering: Advances in Algorithm, Theory, and Applications. CRC Press (2008)
21. Kulis, B., Jordan, M.I.: Revisiting k-means: New algorithms via Bayesian nonparametrics (2011) Arxiv:1111.0352.