Domain Adaptive Classification

Fatemeh Mirrashed [*], Mohammad Rastegari[*]
Department of Computer Science, University of Maryland
College Park, MD, USA
fatemeh,mrastega@cs.umd.edu

Abstract

We propose an unsupervised domain adaptation method that exploits intrinsic compact structures of categories across different domains using binary attributes. Our method directly optimizes for classification in the target domain. The key insight is finding attributes that are discriminative across categories and predictable across domains. We achieve a performance that significantly exceeds the state-of-the-art results on standard benchmarks. In fact, in many cases, our method reaches the same-domain performance, the upper bound, in unsupervised domain adaptation scenarios.

1. Introduction

Discriminative learning algorithms rely on the assumption that models are trained and tested on the data drawn from the same marginal probability distribution. In real world applications, however, this assumption is often violated and results in a significant performance drop. For example, in visual recognition systems, training images are obtained under one set of lighting, background, view point and resolution conditions while the recognizer could be applied to images captured under another set of conditions. In speech recognition, acoustic models trained by one speaker need to be used by another. In natural language processing, part-of-speech taggers, parsers, and document classifiers are trained on carefully annotated training sets, but applied to texts from different genres or styles where there is mismatched distributions of words and their usages.

For these reasons domain adaptation techniques have received considerable attention in machine learning applications. Some previous efforts [24, 3, 8, 7] consider *semi-supervised* domain adaptation where some labeled data from the target domain is available. We focus on the *unsupervised* scenarios when there is no labeled data from the target domain available. Some earlier work in unsupervised domain adaptation assumes that there are discriminative "pivot" features that are common to both domains [5, 4]. While such methods might work well in language domains, in visual world typical histogram-based image descriptors (visual words) can change significantly across domains. A recent work [14] considers the labeled source data at the instance level to detect a subset of them (landmarks) that could model the distribution of the data in the target domain well. A drawback of such methods is that they do not use the information from all the samples in the source domain available for training the classifier, as they use only landmark points and prune the rest.

Another research theme in domain adaptation is to assume there is an underlying common subspace [21, 1, 15] where the source and target domains have the same (or similar) marginal distributions, and the posterior distributions of the labels are also the same across domains. Hence, in this subspace a classifier trained on the labeled data from the source domain would likely perform well on the target domain. However, transforming data only with the goal of modeling the target domain distribution does not necessarily result in accurate classification. Our goal is to identify a transformation that not only models the distribution of a target domain, but also is discriminative across categories.

We propose a simple yet effective adaptation approach that directly learns a new feature space from the unlabeled target data. This feature space is optimized for classification in the target domain. Motivated by [22], our new feature space, composed of binary attributes, is spanned by max-margin non-orthogonal hyperplanes learned directly on the target domain. Our new binary feature sets are discriminative and at the same time are robust against the change of distributions of data points in the original feature space between the source and target domains. We refer to this property as *predictability*. The notion of predictability is based on the idea that subtle variations of the data point positions in the original space should not result in different binary codes. In other words, a particular bit in the binary code should be identical (predictable) for all the data samples that are close to each other in the feature space. Figure 1

[*]The authors contributed equally to this work.

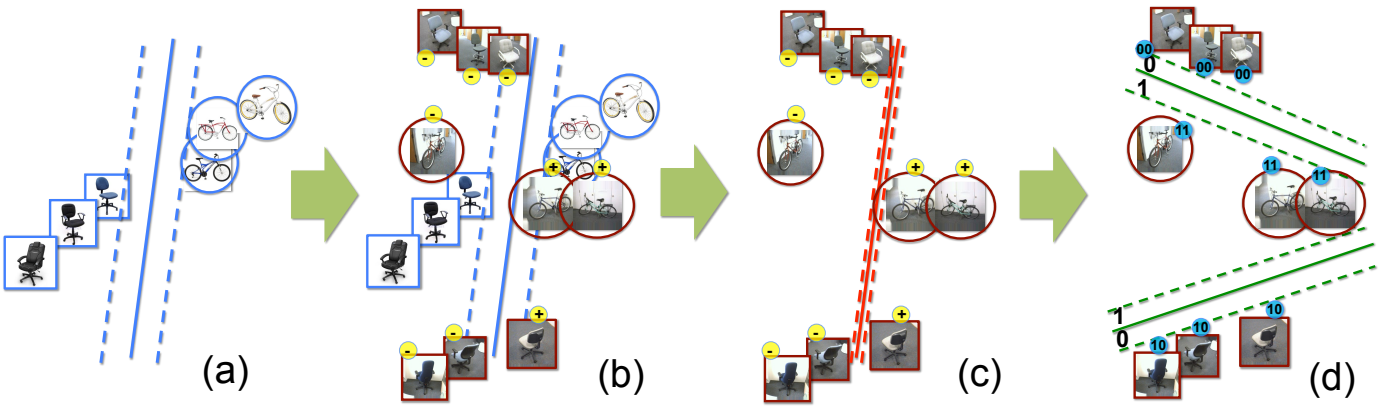

Figure 1. This figure summarizes the overall idea of our method. (a) shows a classifier that is trained on data for two categories from the **source** domain (internet images). In (b) we classify the data from the **target** domain (webcam images) using the classifier trained in (a). In (c) and (d) we want to use roughly predicted labels in the target domain to find hyperplanes that are discriminative across categories and also have large margins from samples. (c) illustrates a hyperplane that perfectly separates positive and negative samples but has a small margin. (d) shows two hyperplanes that are not perfectly discriminative but they are binarizing data in the target domain with a large margin. The binarized samples by these two hyperplanes are linearly separable.

illustrates the essential idea behind our approach.

Our experimental evaluations show that our method significantly outperforms state-of-the-art results on several benchmark datasets which are extensively studied for domain adaptation. In fact in many cases we even reach the upperbound accuracy that is obtained when the classifier is trained and tested on the target domain itself. We also investigate the dataset bias problem, recently studied in [25, 18]. We show that our adaptive classification technique can successfully overcome the bias differences between the datasets in cross-dataset classification tasks. The joint optimization criteria of our model can be solved efficiently and is very easy to implement. Our MATLAB code is online available[1].

2. Related Work

While it is still not clear how exactly to quantify a domain shift between the train (source) and test (target) data sets, several methods have been devised that show improved performance for cross-domain classification.

In language processing, Daume et al [6] model the data distribution corresponding to source and target domains as a common shared component and a component that is specific to the individual domains. Blitzer et al [5, 4] proposed a structural correspondence learning approach that detects some pivot features that occur frequently and behave similarly in both domains. They used these pivot features to learn an adapted discriminative classifier for the target domain. In visual object recognition, Saenko et al [24] proposed a metric learning approach that uses labeled data in the source and target domains for all or some of the corresponding categories to learn a regularized transformation for mapping between the two domains.

In unsupervised settings where there is no label information available from the target domain, several methods have been recently proposed. Pan et al [21] devise a dimensionality reduction technique that learns an underlying subspace where the difference between the data distributions of the two domains is reduced. However they obtain this subspace by aligning distribution properties that are not class-aware; therefore it does not guarantee that the same class from separate domains will project onto the same coordinates in the shared subspace. Gopalan et al [1] take an incremental learning approach, following a geodesic path between the two domains modeled as points on a Grassmann manifold. Gong et al [15] advance this idea by considering a kernel-based approach; i.e. they integrate an infinite number of subspaces on that geodesic path rather than sampling a finite number of them. In [14], Gong et al, however, consider only a subset of training data in the source domain for their geodesic flow kernel approach; the ones that are distributed similarly to the target domain, .

In [25, 18], the varying data distribution between the train and test sets have been studied under the ”dataset bias” They point out how existence of various types of bias, such as capture and negative set bias, between datasets can hurt visual object categorization. This is a similar problem to domain adaptation where each dataset can be considered as a domain.

Another set of related methods are those that use binary code descriptors for recognition. Farhadi et al [11, 12] used binary feature for supervised transfer learning. Recent method shows that even with a few bits of binary descriptor one can reach state-of-the-art performance in object recog-

[1] http://www.umiacs.umd.edu/ mrastega/paper/dom.zip

nition. Gong et al [16] optimized to find a rotation of data that minimizes binary quantization error. They used CCA in order to leverage labels' information. In [22] they proposed a technique to map the data into a hamming space where each bit is predictable from neighboring visual data. At the same time the binary code of an image needs to be discriminative across the categories. Our method is motivated by their approach. We are also looking for a set of discriminative binary codes but in our problem data comes from different domains with mismatched distributions in the feature space. In section 3 we explain how our method solves this problem by a joint optimization over solving a linear SVM and finding a binary projection matrix.

3. Proposed Method

Our goal is to identify useful information for classification in the target domain. We represent this information by a number of hyperplanes in the feature space created using data from the target domain. We call each of these hyperplanes, an *attribute*. These attributes must be discriminative across categories and predictable across domains. We explain our notion of predictability in section 3.2. We use these attributes as feature descriptors and train a classifier on the labeled data in the source domain. When we apply this classifier to the target domain, we achieve a much higher accuracy rate than the baseline classifier for the target data. The baseline is simply a classifier trained on the source data in the original feature space.

Each attribute is a hyperplane in feature space; it divides the space into two subspaces. We assign a binary value to each instance by its "sidedness" with respect to the hyperplane. We construct a K-bit binary code for each image using K hyperplanes. To produce consistent binary codes across domains, each binary value needs to be predictable from instances across domains. Predictability is the key to the performance of our method. We also want the attributes to be discriminative across categories. i.e. the K-bit attribute descriptors of the samples from same category should be similar to each other and different from the other categories.

3.1. Problem Description

First we explain the notations that we use throughout this section. Superscripts $\mathcal{S}$ and $\mathcal{T}$ indicates source and target domains respectively and superscript T indicates matrix transpose. x_i is a d-dimensional column vector that represents the i^{th} instance feature and X is a matrix created by concatenation of all x_i's. l_i is the category label of the i^{th} instance. Without loss of generality, we assume that $l_i \in \{1, -1\}$. A is a $d \times K$ matrix whose k^{th} column, a_k, is the normal vector of a hyperplane (attribute) in the original feature space. w is the K-dimensional normal vector of a classifier that classifies one category from the others in the binary attribute space. sgn$(.)$ is the sign function

We want to directly optimize for better classification in the target domain. Therefore, we need to find K hyperplanes, a_k, in the target domain such that when we use sgn$(A^T x_i)$ as a new feature space, and learn a classifier on source data projected onto this space, we can predict the class labels of the data in the target domain. Of course we do not have the class labels for the data in the target domain $l_i^{\mathcal{T}}$. In order to train the classifier and attributes (hyperplanes) in target domain, we add a constraint to our optimization to force the $l_i^{\mathcal{T}}$ to be predictable from the source domain's classifier. More specifically, our optimization is a combination of two max-margin SVM-like classifiers that are interconnected via the attribute mapping matrix A.

$$\min_{A, w^{\mathcal{S}}, w^{\mathcal{T}}, l^{\mathcal{T}}, \xi^{\mathcal{S}}, \xi^{\mathcal{T}}} \|w^{\mathcal{S}}\| + \|w^{\mathcal{T}}\| + C_1 \sum \xi^{\mathcal{S}} + C_2 \sum \xi^{\mathcal{T}}$$
$$s.t.$$
$$l_i^{\mathcal{S}} ({w^{\mathcal{S}}}^T \text{sgn}(A^T x_i^{\mathcal{S}})) > 1 - \xi_i^{\mathcal{S}},$$
$$l_j^{\mathcal{T}} ({w^{\mathcal{T}}}^T \text{sgn}(A^T x_j^{\mathcal{T}})) > 1 - \xi_j^{\mathcal{T}},$$
$$l_j^{\mathcal{T}} = \text{sgn}({w^{\mathcal{S}}}^T \text{sgn}(A^T x_j^{\mathcal{T}})), \tag{1}$$

It is not straightforward to solve the optimization in Eq 1 because matrix A in the constraints requires a combinatorial search for the optimal solution. But if we constrain the possible solutions for A, then we can solve it efficiently. As we will explain in section 3.2, we do this by forcing predictability constraints on all the a_k vectors.

3.2. Predictability

In different domains data appears with different distributions. Consider a picture of a car taken by a mobile phone's camera and the same picture taken from a professional high quality camera. Due to differences in the two photo capturing systems such as resolution, the two images will be mapped to two different points in visual feature space despite being the same object from the same category. For better classification, however, ideally we would like to create a feature space that would map these two images onto the same or nearby points. In other words, we would like to have a class-compact and domain-invariant feature space for these images. For a sample, an attribute is a binary value derived from a hyperplane in the raw feature space. If this hyperplane produces different binary values for samples that are nearby to each other, then we say that the values coming from this hyperplane are not predictable. Therefore, this attribute would not be robust against the variations of samples from different domains in the raw feature space.

Predictability is the ability to predict the value of a given bit of a sample by looking at the corresponding bit of the

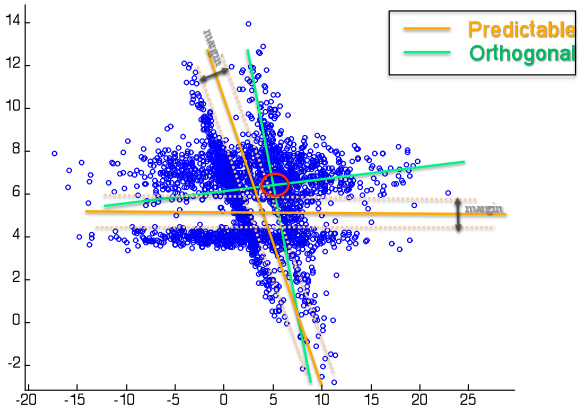

Figure 2. Comparison of predictable hyperplanes and orthogonal hyperplanes. Note that the hyperplanes learned by large margin divide the space, avoiding the fragmentation of sample distributions by the help of *predictability* constraints implemented by max-margin regularization.

nearest neighbors of that sample. For example, if the k^{th} bit in most of the nearest neighbors of a sample is **1** then we can infer that the k^{th} bit of that sample would also be **1**.

Consider the situation where a hyperplane crosses a dense area of samples. There would be many samples in proximity to each other that are assigned different binary values. The binary values obtained by this hyperplane are thus not *predictable*. The binary values obtained by a hyperplane are *predictable* when the hyperplane has large margin from samples. There are several methods that try to model the transfer of distribution between domains [1, 15, 21]. All of these methods rely on discovering some orthogonal basis of the feature space such as principle components. However these orthogonal basis are not appropriate as hyperplanes for attributes. Figure 2 illustrates a demonstration of the hyperplanes defined by orthogonal basis (PCA) in green lines. Note that PCA hyperplanes cross dense areas of samples. If we binarize the samples by the PCA hyperplanes, then samples in the red circle will have different binary codes even though they are nearby each other and strongly clustered. The hyperplanes that are shown in orange are our predictable attributes, which enforce the large margins from samples.

To enforce the predictability constraint on binary values of attributes, we regulate our optimization by adding a max-margin constraint on A as follows:

$$\min_{A,w^{\mathcal{S}},w^{\mathcal{T}},l^{\mathcal{T}},\xi^{\mathcal{S}},\xi^{\mathcal{T}},\xi^{A}} \|w^{\mathcal{S}}\| + \|w^{\mathcal{T}}\| + \|A\|_F + C_1 \sum \xi^{\mathcal{S}} + C_2 \sum \xi^{\mathcal{T}} + C_3 \sum \xi^{A}$$

$$s.t.$$

$$\begin{aligned} & l_i^{\mathcal{S}}({w^{\mathcal{S}}}^T \text{sgn}(A^T x_i^{\mathcal{S}})) > 1 - \xi_i^{\mathcal{S}}, \\ & l_j^{\mathcal{T}}({w^{\mathcal{T}}}^T \text{sgn}(A^T x_j^{\mathcal{T}})) > 1 - \xi_j^{\mathcal{T}}, \\ & l_j^{\mathcal{T}} = \text{sgn}({w^{\mathcal{S}}}^T \text{sgn}(A^T x_j^{\mathcal{T}})), \\ & b_{kj} = \text{sgn}(a_k^T x_j^{\mathcal{T}}), \\ & b_{kj}(a_k^T x_j^{\mathcal{T}}) > 1 - \xi_{kj}^{A}, \end{aligned} \tag{2}$$

Where b_{kj} is the binary value of the k^{th} bit (attribute) of the j^{th} sample in the target domain. In fact, each attribute is a max-margin classifier in feature space and b_{jk} is the label of the j^{th} sample when classified by the k^{th} attribute classifier. This optimization can be easily conducted using block coordinate descent. If we fix $w^{\mathcal{T}}$ and A, then solving the optimization for $w^{\mathcal{S}}$ is a simple linear SVM in the attribute space. Accordingly, once we determine $w^{\mathcal{S}}$, we can compute $l^{\mathcal{T}}$. Then solving for $w^{\mathcal{T}}$ and A is a standard attribute discovery problem in the target domain and can be solved using the method (DBC) in [22]. We iterate over these two steps: finding $w^{\mathcal{S}}$, and then solving for $w^{\mathcal{T}}$ and A. We don't know how to obtain a good initialization for $w^{\mathcal{T}}$ and A, but luckily we don't necessarily need them. We only need to have an initialization for $l^{\mathcal{T}}$ so that we can solve the attribute discovery problem for A and $w^{\mathcal{T}}$. An intuitive way to initialize $l^{\mathcal{T}}$ is to learn a classifier on the labeled data in the source domain, $x^{\mathcal{S}}$ and $l^{\mathcal{S}}$, and then apply it on $x^{\mathcal{T}}$, the data in the target domain. Algorithm 1 summarizes our method.

Algorithm 1 Adaptive Classification

Input: $X^{\mathcal{S}}, l^{\mathcal{S}}, X^{\mathcal{T}}, K$.
Output: $l^{\mathcal{T}}, A, w^{\mathcal{S}}, w^{\mathcal{T}}$.

```
θ ← Learn a classifier on X^S and l^S
l^T ← Test the classifier θ on X^T //Initialization for l^T
repeat
    w^T, A ← DBC(X^T, l^T, K)
    w^S ← Learn a linear SVM on sgn(A^T X^S) and l^S
    l^T ← sgn(w^S^T sgn(A^T X^T))
until convergence on l^T
```

4. Experiments

We first evaluate our method on two benchmark datasets extensively used for domain adaptation in the contexts of object recognition [24, 19, 1, 15, 14] and sentiment analysis [4, 1, 14]. We compare our method to several previously published domain adaptation methods. Empirical results show that our method not only outperforms all prior

techniques in almost all cases, but also in many cases we achieve the same-domain classification, the upper bound, accuracy, i.e. when the classifier is trained and tested on the target domain itself.

Furthermore, we test the performance of our method on an inductive setting of unsupervised domain adaptation. In the inductive setting we test our adapted classifier on a set of unseen and unlabeled instances from target domain- separate from the target domain data used to learn the attribute model. And finally, we investigate the dataset bias problem, recently studied in [25, 18], and we show that our adaptive classification technique can successfully overcome the bias differences in both single and multiple source domains scenarios.

4.1. Cross-Domain Object Recognition

First, we evaluate our method for cross-domain object recognition. We followed the setup of [15, 14] which use the three datasets of object images studied in [24, 19, 1]: Amazon (***A***) (images downloaded from online merchants), Webcam (***W***) (low-resolution images taken by a web camera), and DSLR (***D***) (high-resolution images taken by a digital SLR camera) plus Caltech-256 (***C***) [13]as a fourth dataset. Each dataset is regarded as a domain. The domain shift is caused by factors including change in resolution, pose, lighting, background, etc. The experiments are conducted on 10 object classes common to all 4 datasets. There are 2533 images in total and the number of images per class ranges from 15 (in DSLR) to 30 (Webcam), and up to 100 (Caltech and Amazon). We used the publicly available feature sets [2], and the same protocol as in all the previous work were used for representing images: The 64-dimensional SURF features [2] were extracted from the images, and a codebook of size 800 was generated by k-means clustering on a random subset of Amazon database. Then, the images from all domains are represented by an 800-bin normalized histograms corresponding to the codebook.

We report the results of our evaluation on all 12 pairs of source and target domains and compare it with methods as reported in [14] (Table 1). The other methods include transfer component analysis (tca) [21], geodesic flow sampling (gfs) [1], Geodesic Flow Kernel (gfk)[15], structural correspondence learning (scl)[5], kernel mean matching (kmm) [17], and a metric learning method (metric) [24] for semi-supervised domain adaptation, where label information (1 instance per category) from the target domains is used. We also report a baseline results of no adaptation, where we train a kernel SVM on labeled data from the source domain in the original feature space. A linear kernel function is used for the SVM. For each pair of domains the performance is measured by classification accuracy (number of correctly classified instances over total test data from target).

[2]http://www-scf.usc.edu/ boqinggo/da.html

	$K \rightarrow D$	$D \rightarrow B$	$B \rightarrow E$	$E \rightarrow K$
No Adaptation	72.7	73.4	73	81.4
TCA [21]	60.4	61.4	61.3	68.7
GFS [1]	67.9	68.6	66.9	75.1
GFK [15]	69.0	71.3	68.4	78.2
SCL [5]	72.8	76.2	75.0	82.9
KMM [17]	72.2	78.6	76.9	83.5
Metric [24]	70.6	72.0	72.2	77.1
Landmark [14]	75.1	79.0	78.5	83.4
Ours	**92.1**	**93.15**	**94.94**	**95.65**

Table 2. **Cross-Domain Sentiment Classification:** accuracies for 4 pairs of source and target domains are reported. K: kitchen, D: dvd, B: books, E: electronics. Our method outperforms all the previous methods.

As explained in [14], due to its small number of samples (157 for all 10 categories), DSLR was not used as a source domain and so the results for other methods have been reported only for 9 out of 12 pairings. Table 1 shows that our method outperforms all the previous methods in all cases except when DSLR is the target domain. The culprit is the small number of samples in DSLR being insufficient for training the attribute model. In all our experiments in this paper, we used a binary attribute space with 256 dimensions. To learn each attribute hyperplane we used linear SVM coupled with kernel mapping. None of the hyperparameters for SVM classifiers and DBC model were tuned. They were all left at their default values. One might get better results by tuning these parameters.

4.2. Cross-Domain Sentiment Analysis

Next, we consider the task of cross-domain sentiment analysis in text [4]. Again we compare the performance of our approach with the same set of domain adaptation methods as reported in [14] and listed in section 4.1. We used the dataset in [4] which includes product reviews from amazon.com for four different products: books (***B***), DVD (***D***), electronics (***E***), and kitchen appliances (***K***). Each product is considered as a domain. Each review has a rating from 0 to 5, a reviewer name and location, review text, among others. Reviews with rating higher than 3 were classified as positive, and those less than 3 were classified negative. The goal is to determine whether the process of learning positive/ negative reviews from one domain, is applicable to another domain. We used the publicly available feature sets for the collection in which bag-of-words features are used and the dimensionality of data is reduced to 400 (the 400 words with the largest mutual information with the labels).

Table 2 shows the results; our method outperforms all the previous methods by a relatively large margin (25% average improvement over baseline and 19% over state-of-art).

	$A \to C$	$A \to D$	$A \to W$	$C \to A$	$C \to D$	$C \to W$	$W \to A$	$W \to C$	$W \to D$	$D \to W$	$D \to C$	$D \to A$
No Adaptation	41.7	41.4	34.2	51.8	54.1	46.8	31.1	31.5	70.7	38.2	34.6	38.2
TCA [21]	35.0	36.3	27.8	41.4	45.2	32.5	24.2	22.5	80.2	N/A	N/A	N/A
GFS [1]	39.2	36.3	33.6	43.6	40.8	36.3	33.5	30.9	75.7	N/A	N/A	N/A
GFK [15]	42.2	42.7	40.7	44.5	43.3	44.7	31.8	30.8	75.6	N/A	N/A	N/A
SCL [5]	42.3	36.9	34.9	49.3	42.0	39.3	34.7	32.5	83.4	N/A	N/A	N/A
KMM [17]	42.2	42.7	42.4	48.3	53.5	45.8	31.9	29.0	72.0	N/A	N/A	N/A
Metric [24]	42.4	42.9	49.8	46.6	47.6	42.8	38.6	33.0	**87.1**	N/A	N/A	N/A
Landmark [14]	45.5	47.1	46.1	56.7	**57.3**	49.5	40.2	35.4	75.2	N/A	N/A	N/A
Ours	**75.15**	**51.59**	**52.54**	**91.54**	49.68	**60.34**	**74.22**	**53.34**	76.43	**81.02**	**56.03**	**72.03**

Table 1. **Cross-domain Object recognition:** accuracies for all 12 pairs of source and target domains are reported (C: Caltech, A: Amazon, W: Webcam, and D: DSLR). Due to its small number of samples, DSLR was not used as a source domain by the other methods and so their results have been reported only for 9 pairings. Our method significantly outperforms all the previous methods except for 2 out of 3 cases when DSLR , whose number of samples are insufficient for training our attribute model, is the target domain.

4.3. Comparing to Same-Domain Classification

How accurate are the domain adapted classifiers compared to classifiers trained on labeled data from the target domain? To investigate this, we divide each dataset into two equal parts, one of which is used for training and the other for testing. This balances the number of samples used for within domain training and testing and cross domain adaptive training and testing.

Table 3 shows the results for all 16 pairs of domains in sentiment dataset and 4 pairs of domains from object recognition datasets. In the latter we could use only the two domains (Caltech, Amazon) that had sufficient number of samples to be divided into two groups (train/test)

The rows correspond to the source domains and columns to the target domains. We can see how on this data set our adaptive classification method reaches the upper bound performance in all cases.

	K	E	B	D
K	97.9	97.4	96.6	95.2
E	97.9	97.4	96.5	95.4
B	97.8	97.4	96.6	95.3
D	97.7	97.3	96.6	95.4

	C	A
C	75.6	92.2
A	74.4	92.2

Table 3. **Comparing to Same-Domain Classification :** (Left) Accuracies for all 16 pairs of source and target domains in sentiment dataset are reported in the left table. K: kitchen, D: dvd, B: books, E: electronics. (Right) Accuracies for 4 pairs of source and target domains are reported. C: Caltech, A: Amazon. Rows and columns correspond to source and target domains respectively. Our method reaches the upper bound accuracies (diagonal) for cross-domain classification.

4.4. Transductive vs Inductive Cross-Domain Classification

In the previous experiments, we follow the same protocol as [15, 14] for a fair comparison. So, we had access to all the samples in the target domain at training time and our goal was to predict their labels. This is a transductive learning problem except that the test data was drawn from a different domain. In an inductive setting we do not have access to the test data at training time. So, to create an inductive setting for the unsupervised domain adaptation problem, we make only a fraction of the data from the target domain accessible at training time for learning our adaptive feature space. The rest, which we refer to as out-of-sample data from the target domain, is set aside for inductive classification tests.

Table 4, reports the results for this experiment on the sentiment data set where we have balanced number of samples across domains. Our adaptive classification results on out-of-sample data still outperform the corresponding performance for in-sample data by other methods in 3 out of 4 cases. Nevertheless, it does show a drop in performance compared with our own in-sample results. As we show later, however, this is not necessarily the case. In section 4.5 we show how our out-of-sample results reasonably perform compared to the corresponding in-sample ones. (table 5)

		$K \to D$	$D \to B$	$B \to E$	$E \to K$
In-samples	No Adaptation	72.7	77.1	75.2	82.8
	Adapted (Ours)	**97.2**	**96.6**	**98.0**	**98.1**
Out-samples	No Adaptation	70.5	75.6	74.4	82.8
	Adapted (Ours)	**77.5**	**76.9**	**80.7**	**84.4**

Table 4. **Transductive vs Inductive Cross-domain Classification:** The first two rows show the results in transductive setting where all the data from the target domains are accessible during training. The last two rows show the results in inductive setting where we test our classifier only on a subset of data in the target domain that was not accessible during training time

4.5. Dataset Bias

Most of the images in the datasets studied in sections 4.1 and 4.2 contain the object of interest centered and cropped on a mostly uniform background. To evaluate our method on a wider range of images with unconstrained backgrounds and clutter, as well as to see how it deals with the data set bias problem addressed in [25, 18], we extend our cross-domain object recognition experiments to four widely used

		Caltech	LabelMe	Pascal07	SUN09
In-samples	No Adaptation	78.7	71.6	76.1	70.9
	Adapted (Ours)	**99.4**	**92.7**	**92.6**	**94.9**
Out-samples	No Adaptation	79.1	75.1	75.0	74.2
	Adapted (Ours)	**94.6**	**86.4**	**90.1**	**87.8**

Table 5. **Cross-Dataset Object Recognition:** The 4 rightmost columns show the classification results for when we hold out one dataset as the target domain and use the other 3 as source domains, in both the inductive (first two rows) and transductive (last two rows) settings. The reported results are averaged over 5 categories of objects.

computer vision datasets- Pascal2007 [9], SUN09 [27], LabelMe [23], Caltech101 [13].

We follow the same protocol as [18], where they run experiments on five common object categories- "bird", "car", "chair", "dog", and "person". We used the publicly available feature sets for this data [3]. Using a bag-of-words representation, Grayscale SIFT descriptors [20] at multiple patch sizes of 8, 12, 16, 24 and 30 with a grid spacing of 4 were extracted. Using k-means clustering on randomly sampled descriptors from the training set of all datasets, a codebook of size 256 is constructed. The baseline SVM is implemented using Liblinear [10] coupled with a Gaussian kernel mapping function [26]. The results are evaluated by average precision (AP).

Table5 reports the results of our cross-dataset classification in both the inductive (in-sample) and transductive (out-of-sample) settings. Each column of the table correspond to the situation where one dataset is considered as the target domain and all the remaining datasets are considered as the source domain (multi-source domain). These result shows that our approach is robust against varying biases when the training data comes from multiple datasets and the test data comes from another one. The reported results are averaged over all 5 categories. The average performance improvement by our adaptive method over the baseline (no adaptation) is 28% for out-of-sample data and 18% for in-sample data. The only related work that we are aware of that has performed theses cross-dataset classifications experiments with the same settings is [18] where they report an average performance improvement of only 2.5% across all datasets and all categories.

4.6. Effectiveness of Predictability

Now, we show the importance of the predictability of attributes by quantitative and qualitative evaluations.

Quantitative evaluation: To see how learning binary attributes by itself is contributing to our performance increase, we ignore the adaptation and use the attribute features learned only from the source domain. In this setting we learn the binary attribute space from the labeled data in the source domain, project the data from both source and target domain onto this space where we train a classifier on the source data and test it on the target data. We then compare the results with corresponding ones by our adapted model. We used the same experiment setup in section 4.5 for this evaluation (Figure 3).

[3] http://undoingbias.csail.mit.edu/features.tar

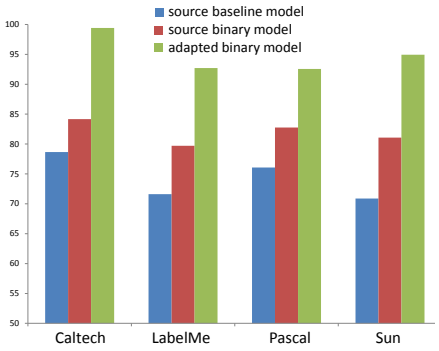

Figure 3. **Quantitative Evaluation of Predictability:** The blue bars show the classification accuracies when the classifier is simply trained on the data from the source domain in original feature space (baseline). The red bars show the results when the classifier is trained in a binary attribute space learned from the data in the source domain (source binary). The green bars show the results of our adapted model when the classifier is trained on labeled source data in a binary attribute space learned in the target domain (adapted binary). In average the source binary model is increasing the performance by 10% over the baseline while the adapted binary model does that by 28%

Qualitative evaluation: Here we show that the discovered attributes are consistent across domains. We pick an attribute classifier learned by our method, then we find images (from both source and target) that are most positively and negatively confident when classified by this attribute classifier. In Figure 4 the left two rows use DSLR as source domain and Amazon as target. Similarly, the right two rows use Amazon as source and Webcam as target. The green arrow represent an attribute classifier which is trained on target domain. The dashed part of the arrow illustrates that the same hyperplane which is trained in target domain is applied in the source domain. Images on the right side of the green arrow are the most positive and on the left side are the most negative one. As can be seen in both cases the attribute classifiers are consistent across domains. In the first case, the attribute consistently separates round shapes from dark-volumed shapes in both domains and in the second case, the attribute consistently discriminates between objects with keypad and objects with dark-volumed shape. This observation is consistent with our intuition of predictability in our optimization.

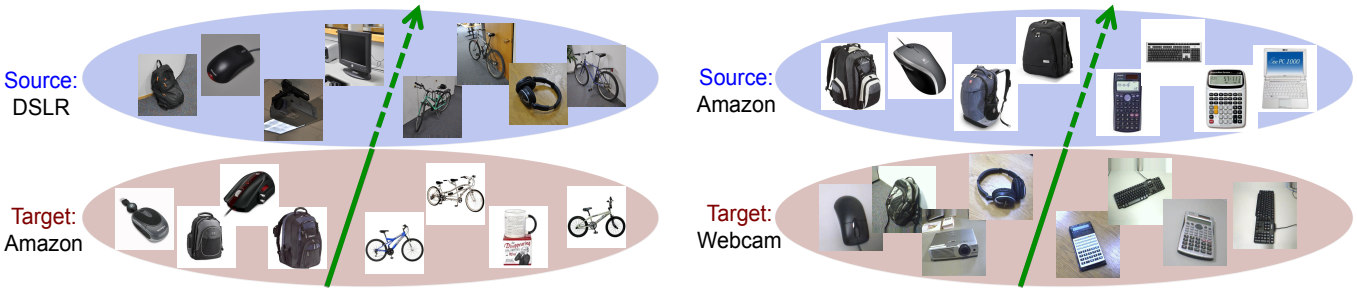

Figure 4. **Qualitative Evaluation of Predictability:** This figure illustrates two examples where an attribute hyperplane (green arrow), learned by our joint optimization, discriminates visual properties consistently across two different domains. In the left case, the hyperplane is discriminating between the objects with round shapes vs the ones with more surface area. In the right example, the hyperplane is discriminating the keypad-like objects against the more bulky ones. The dashed part of the arrow indicates that the same hyperplane which is trained in target domain is applied in the source domain.

5. Conclusion

We introduce a method for adaptive classification when the train and test data come from different domains. Our method is based on learning a predictable binary code that captures the structural information of the data distribution in the target domain itself. These binary codes prove to be highly effective for classification since they are optimized to be robust against the variations of data distribution in the feature space, while they maintain their discriminative properties. We designed a joint optimization that learns both binary projection matrix and the classifier and is very easy to implement.

Our empirical evaluations demonstrate an impressive and consistent performance gain by our method on standard benchmarks previously studied for domain adaptation problem. In many cases our domain adaptive method could reach the gold standard accuracies; i.e. when the classifier is trained on the labeled from the target domain itself. We also show how our method can successfully generalize over the bias variations among widely-used computer vision datasets.

Acknowledgment: This work was partially supported by MURI from the Ofce of Naval Research under the Grant N00014-10-1-0934.

References

[1] R. G. ad R. Li and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, 2011. 1, 2, 4, 5, 6

[2] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *In ECCV*, 2006. 5

[3] A. Bergamo and L. Torresani. Exploiting weakly-labeled web images to improve object classification: A domain adaptation approach. In *NIPS*, 2010. 1

[4] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007. 1, 2, 4, 5

[5] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Conference on Empirical Methods in Natural Language Processing*, 2006. 1, 2, 5, 6

[6] H. Daumé, III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 2006. 2

[7] H. Daume III. Frustratingly easy domain adaptation. 2007. 1

[8] H. Daume III, A. Kumar, and A. Saha. Co-regularization based semi-supervised domain adaptation. In *NIPS*, 2010. 1

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88, 2010. 7

[10] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 2008. 7

[11] A. Farhadi, D. A. Forsyth, and R. White. Transfer learning in sign language. In *CVPR*, 2007. 2

[12] A. Farhadi and M. K. Tabrizi. Learning to recognize activities from the wrong view point. In *ECCV*, 2008. 2

[13] A. H. G. Griffin and P. Perona. Caltech-256 object category dataset. In *Technical report*, 2007. 5, 7

[14] B. Gong, Y. Shi, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013. 1, 2, 4, 5, 6

[15] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012. 1, 2, 4, 5, 6

[16] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, 2011. 3

[17] H. J., A. Gretton, K. Borgwardt, and B. Scholkopf. Correcting sample selection bias by unlabeled data. In *NIPS*, 2007. 5, 6

[18] A. Khosla, T. Zhou, T. Malisiewicz, A. Efros, and A. Torralba. Undoing the damage of dataset bias. In *ECCV*, 2012. 2, 5, 6, 7

[19] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, 2011. 4, 5

[20] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 2, 2004. 7

[21] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2), 2011. 1, 2, 4, 5, 6

[22] M. Rastegari, A. Farhadi, and D. Forsyth. Attribute discovery via predictable discriminative binary codes. In *ECCV*, 2012. 1, 3, 4

[23] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. *IJCV*, 2007. 7

[24] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *ECCV*, 2010. 1, 2, 4, 5, 6

[25] A. Torralba and A. Efros. Unbiased look at dataset bias. In *CVPR*, 2011. 2, 5, 6

[26] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008. 7

[27] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 7