Max-margin transforms for visual domain adaptation

Judy Hoffman
UC Berkeley, EECS Department
jhoffman@eecs.berkeley.edu

Erik Rodner
ICSI
Erik.Rodner@uni-jena.de

Jeff Donahue
UC Berkeley, EECS Department
jdonahue@eecs.berkeley.edu

Kate Saenko
University of Massachusetts, Lowell
ICSI; Harvard
saenko@eecs.berkeley.edu

Trevor Darrell
UC Berkeley, EECS Department; ICSI
trevor@eecs.berkeley.edu

Abstract

We present a new algorithm for training linear support vector machine classifiers across image domains. To compensate for statistical differences between domains, our algorithm learns a linear transformation that maps points from the target (test) domain to the source (training) domain as part of training the classifier. We optimize both the transformation and classifier parameters jointly, and introduce a novel cost function for transformation learning based on the misclassification loss of the target points transformed into the source domain. Our method has advantages over previous SVM-based domain adaptation algorithms because it performs multi-task adaptation, learning a shared component of the domain shift across all categories. Additionally, our method has an advantage over the previous max-margin techniques because it can be solved in linear feature space, making it scalable to large training datasets. We present experiments on both synthetic data and real image datasets that demonstrate strong performance and computational advantages compared to previous approaches.

1 Introduction

We address the problem of adapting image classifiers to novel domains. Recent studies have demonstrated a significant degradation in the performance of state-of-the-art image classifiers due to test domain shifts such as changing image sensors and noise conditions [1], pose changes [2], consumer vs. commercial video [3, 4], and, more generally, training datasets biased by the way in which they were collected [5]. Adaptation of support vector machines is a particularly interesting problem due to their prevalence, with fast linear SVMs forming the core of some of the most popular object detection methods [6, 7].

Several recent SVM adaptation methods have been proposed for vision applications [3, 4, 8, 9, 10, 11]. In particular, adaptive linear SVMs [9, 10, 11] learn a perturbation of the source hyperplane by minimizing the classification error on target labeled examples for each binary task. Figure 1(a) illustrates the adaptation of the source hyperplane parameter θ for a binary *cup* classifier using a small number of labeled target images (green border indicates a positive label, red negative) to obtain the target parameter θ^t.

(a) Adaptive SVM learns a new θ^t using target labels for *cup*

(b) Adaptive SVM cannot learn θ^t without target labels for *phone*

(c) Our method uses a task-independent transform W

Figure 1: Supervised max-margin adaptation techniques such as PMT-SVM [11] cannot generalize the learned domain shift to novel tasks/categories: (a) these methods learn a separate adapted parameter θ^t for each task by minimizing its distance from the source parameter θ learned on the source domain (images without borders) while minimizing the misclassification of labeled images from the target domain (green border is positive, red is negative); (b) given a novel task with no labels in the target, these methods cannot predict the adapted parameter θ^t for that task; (c) in contrast, our method learns a single parameter transform W for all tasks/categories and is thus able to obtain adapted parameters $\theta^t = W^T\theta$ for all tasks.

In many vision applications, such as object, face or activity recognition, the number of categories is high and only a few categories have a small number of labels in the target domain. This poses a problem for SVM-based methods, as they are unable to either share the training labels across categories or adapt categories unlabeled in the target, as illustrated in Figure 1(b): without labeled target points for *phone*, adaptation cannot be performed.

Recently proposed transform-based adaptation methods [1, 2, 12, 13, 14] overcome this problem by learning a feature transform that maps target features into the source, pooling all training labels across categories. This enables them to perform multi-task adaptation, and to transfer the task-independent component of the domain shift (the feature transform) to unlabeled categories. For example, a map learned on the labeled *cup* category can be used to map the unlabeled *phone* category to the source domain and then apply a source *phone* classifier. An additional advantage of the asymmetric transform method ARC-t [12] is that it can learn maps between heterogeneous domains.

While attractive, the approach in [12] has two major flaws: First, unlike the SVM parameter adaptation methods above, transformation learning does not optimize the objective function of a strong, discriminative classifier directly; rather, it maximizes some notion of closeness between the transformed target points and points in the source. The second disadvantage is its increased computational complexity due to the high number of constraints, which is proportional to the product of the number of labeled data points in the source and target. This prevents the method from being applied to source domains with large numbers of points.

A recent approach proposed in [**?**] learns a transformation both from source and from target into a latent common space. It seeks to learn the projections while optimizing the classification objective. Since this method requires learning both transforms for an augmented feature space it makes the problem difficult to optimize directly in linear feature space. Therefore, the authors proposed optimizing a kernelized version of their algorithm. However, this solution has limitations as data sources grow.

In this paper, we present a novel technique that combines the strengths of both transform-based and parameter-based methods, which we call Max-Margin Domain Transforms, or MMDT for short. As shown in Figure 1(c), MMDT uses a transform W to map source model parameters to the target $\theta^t = W^T\theta$, learning the transform jointly on all categories for which target labels are available. MMDT provides a way to adapt max-margin classifiers in a multi-task manner, by learning a shared component of the domain shift as captured by the transformation W. Additionally, MMDT can be optimized quickly in linear space, making it a feasible solution for problem settings with a large amount of training data.

The key idea behind our approach is to simultaneously learn both the projection of the source parameters into the target domain and the classifier parameters themselves, using the same classification loss to jointly optimize both W and θ. Thus our method combines the strengths of max-margin learning with the flexibility of the feature transform: because it operates over the input features, it can generalize the learned shift in a way that parameter-based methods cannot. On the other hand, it overcomes the two flaws of the ARC-t method: by optimizing the classification loss directly in the transform learning framework, it can achieve higher accuracy; furthermore, replacing similarity constraints with more efficient hyperplane constraints significantly reduces the training time of the algorithm and learning a transformation directly from target to source allows optimization in linear space.

The main contributions of our paper can be summarized as follows:

- MMDT can be optimized faster than competing methods because it has fewer constraints to satisfy (than [12]) and because it can be optimized in linear feature space, unlike [**?**].
- Experiments show that MMDT in linear feature space outperforms competing methods in terms of multi-class accuracy even compared to previous kernelized methods.
- MMDT learns an asymmetric category independent transformation. Therefore, it can learn adaptation even when the target domain does not have any labeled examples for some categories and when the target and source features are not equivalent.
- Our final iterative solution can be solved using standard QP packages, making MMDT easy to implement.

2 Related Work

Domain adaptation, or covariate shift, is a fundamental problem in machine learning, and has attracted a lot of attention in the machine learning and natural language community, e.g. [15, 16, 17, 18] (see [19] for a comprehensive overview.) It is related to multi-task learning but differs from it in the following way: in domain adaptation problems, the distribution over the features $\mathrm{p}(X)$ varies across domains while the output labels Y remain the same; in multi-task learning or knowledge transfer, $\mathrm{p}(X)$ stays the same (single domain) while the output labels vary (see [19] for more details.) In this paper, we perform multi-task learning *across domains*, i.e. both $\mathrm{p}(X)$ and the output labels can Y change between domains.

Domain adaptation has been gaining considerable attention in the vision community. Several SVM-based approaches have been proposed for image domain adaptation, including: weighted combination of source and target SVMs and transductive SVMs applied to adaptation in [8]; the feature replication method of [16]; Adaptive SVM [9, 10], where the source model parameters are adapted by adding a perturbation function, and its successor PMT-SVM [11]; Domain Transfer SVM [3], which learns a target decision function while reducing the mismatch in the domain distributions; and a related method [4] based on multiple kernel learning. In the linear case, feature replication [16] can be shown to decompose the learned parameter into $\theta = \hat{\theta} + \theta'$, where $\hat{\theta}$ is shared by all domains [20], in a similar fashion to adaptive SVMs.

Several transform-based adaptation methods [1, 12, 13, 2, 14, **?**] have also recently been proposed for (semi-)supervised visual domain adaptation. These methods attempt to learn a perturbation over the feature space rather than a class-specific perturbation over the model parameters, typically in the form of a transformation matrix.

The ARC-t method [12] learns a transformation matrix W that maximizes similarity constraints between points in the source and those projected from the target domain using W. The mapped points are then used in a separate classifier. ARC-t has demonstrated the ability to perform multi-task adaptation and handle heterogeneous features; however, it has approximately quadratic dependence on the number of training points and does not optimize classification accuracy directly.

The recent HFA method [**?**] learns a transformation both from the source and target into a common latent space where classification can occur. However, this method is limited because it requires learning a transformation for augmented feature spaces which makes it difficult to solve the optimization problem in linear feature space. Therefore, the authors present and implement a kernelized

version of their algorithm. This solution, in turn becoming limiting as the number of training points grows.

In this paper, we incorporate the transformation learning directly into the classification objective and learn it by optimizing the accuracy on the training data. We are able to run our algorithm in linear space and so have the potential to scale up to a large number of training examples.

3 Max-Margin Domain Transforms

We propose a novel method for multi-task domain adaptation of linear SVMs. Denote the normal to the affine hyperplane associated with the k'th binary SVM as θ_k, $k = 1, ..., K$, and the offset of that hyperplane from the origin as b_k. Intuitively, we would like to learn a joint perturbation over θ_k that is shared across multiple categories. We propose to do so by learning a transformation W of the input features, or, equivalently, a transformation W^T of the source hyperplane parameters θ_k. Let $x_1^s, \ldots, x_{n_S}^s$ denote the training points in the source domain ($\mathcal{D}_S$), with labels $y_1^s, \ldots, y_{n_S}^s$. Let $x_1^t, \ldots, x_{n_T}^t$ denote the labeled points in the target domain ($\mathcal{D}_T$), with labels $y_1^t, \ldots, y_{n_T}^t$. Thus our goal is to jointly learn 1) affine hyperplanes that separate the classes in the common domain consisting of the source domain and target points projected to the source and 2) the transformation from the points in the target domain into the source domain. The transformation should have the property that it projects the target points onto the correct side of each source hyperplane.

For simplicity of presentation, we first show the optimization problem for a binary problem (dropping k) with no slack variables. Our objective is as follows:

$$\min_{W,\theta,b} \quad \frac{1}{2}||W||_F^2 + \frac{1}{2}||\theta||_2^2 \tag{1}$$

$$\text{s.t.} \quad y_i^s \left(\begin{bmatrix} x_i^s \\ 1 \end{bmatrix}^T \begin{bmatrix} \theta \\ b \end{bmatrix} \right) \geq 1 \qquad \forall i \in \mathcal{D}_S \tag{2}$$

$$y_i^t \left(\begin{bmatrix} x_i^t \\ 1 \end{bmatrix}^T W^T \begin{bmatrix} \theta \\ b \end{bmatrix} \right) \geq 1 \quad \forall i \in \mathcal{D}_T \tag{3}$$

Note that this can be easily extended to the multi-class case by simply adding a sum over the regularizers on all θ_k parameters and pooling the constraints for all categories.

The objective function, written as in Equations (1)-(3), is not a convex problem and so is both hard to optimize and is not guaranteed to have a global solution. Therefore, a standard way to solve this problem is to do alternating minimization on the parameters, in our case W and (θ, b). We can effectively do this because when each parameter vector is fixed, the resulting optimization problem is convex.

We begin by re-writing Equations (1)-(3) for the more general problem with soft constraints (slack) and K categories. Let us denote the hinge loss as: $\mathcal{L}(y, x, \theta) = \max\{0, 1 - \delta(y, k) \cdot x^T\theta\}$. We define a cost function

$$\begin{aligned} J(W, \theta_k, b_k) \quad = \quad & \frac{1}{2}||W||_F^2 + \sum_{k=1}^{K} \left[\frac{1}{2}||\theta_k||_2^2 \right. \\ & + C_S \sum_{i=1}^{n_S} \mathcal{L}\left(y_i^s, \begin{bmatrix} x_i^s \\ 1 \end{bmatrix}, \begin{bmatrix} \theta_k \\ b_k \end{bmatrix} \right) \\ & \left. + C_T \sum_{i=1}^{n_T} \mathcal{L}\left(y_i^t, W \cdot \begin{bmatrix} x_i^t \\ 1 \end{bmatrix}, \begin{bmatrix} \theta_k \\ b_k \end{bmatrix} \right) \right] \end{aligned} \tag{4}$$

where the constant C_S penalizes the source classification error and C_T penalizes the target adaptation error. Finally, we define our objective function with soft constraints as follows:

$$\min_{W,\theta_k,b_k} J(W, \theta_k, b_k) \tag{5}$$

To solve the above optimization problem we perform coordinate descent on W and (θ, b). Our algorithm takes the following form:

1. Set iteration $j = 0$, $W^{(j)} = 0$.
2. Solve the sub-problem $(\theta_k^{(j+1)}, b_k^{(j+1)}) = \arg\min_{\theta_k, b_k} J(W^{(j)}, \theta_k, b_k)$ by solving:

$$\min_{\theta,b} \quad \sum_{k=1}^{K} \left[\frac{1}{2}||\theta_k||_2^2 \right. \tag{6}$$
$$+C_S \sum_{i=1}^{n_S} \mathcal{L}\left(y_i^s, \begin{bmatrix} x_i^s \\ 1 \end{bmatrix}, \begin{bmatrix} \theta_k; \\ b_k \end{bmatrix}\right)$$
$$\left. +C_T \sum_{i=1}^{n_T} \mathcal{L}\left(y_i^t, W^{(j)} \cdot \begin{bmatrix} x_i^t \\ 1 \end{bmatrix}, \begin{bmatrix} \theta_k \\ b_k \end{bmatrix}\right)\right]$$

   Notice, this corresponds to the standard SVM objective function, except that the target points are first projected into the source using $W^{(j)}$. Therefore, we can solve this intermediate problem using a standard SVM solver package.
3. Solve the subproblem $W^{(j+1)} = \arg\min_W J(W, \theta^{(j+1)}, b^{(j+1)})$ by solving

$$\min_{W} \quad \frac{1}{2}||W||_F^2 + \tag{7}$$
$$C_T \sum_{k=1}^{K} \sum_{i=1}^{n_T} \mathcal{L}\left(y_i^t, W \cdot \begin{bmatrix} x_i^t \\ 1 \end{bmatrix}, \begin{bmatrix} \theta_k^{(j+1)} \\ b_k^{(j+1)} \end{bmatrix}\right)$$

   and increment j. This optimization sub-problem is convex and is in a form that a standard QP optimization package can solve.
4. Iterate steps 2 & 3 until convergence.

It is straightforward to show that both stages (2) and (3) cannot increase the global cost function $J(W, \theta, b)$. Therefore, this algorithm is guaranteed to converge to a local optimum. A proof is included in the supplemental material.

It is important to note that since both steps of our iterative algorithm can be solved using standard QP solvers, the algorithm can be easily implemented. Additionally, since the constraints in our algorithm grow linearly with the number of training points and it can be solved in linear feature space, the optimization can be solved efficiently even as the number of training points grows.

We now analyze the proposed algorithm in the context of the previous feature transform methods ARC-t [12] and HFA [**?**]. ARC-t introduced similarity-based constraints to learn a mapping similar to that in step 3 in our algorithm. This approach creates a constraint for each labeled point x_i^s in the source and labeled point x_i^t in the target, and then learns a transformation W that satisfies constraints of the form $(x_i^s)^T W x_i^t > u$ if the labels of x_i^s and x_i^t are the same, and $(x_i^s)^T W x_i^t < l$ if the labels are different, for some constants u, l.

The ARC-t formulation has two distinct limitations that our method overcomes. First, it must solve $n_S \cdot n_T$ constraints, whereas our formulation only needs to solve $K \cdot n_T$ constraints, for a K category problem. In general, our method scales to much larger source domains than with ARC-t. The second benefit of our max-margin transformation learning approach is that the transformation learned using the max-margin constraints is learned jointly with the classifier, and explicitly seeks to optimize the final SVM classifier objective. While ARC-t's similarity-based constraints seek to map points of the same category arbitrarily close to one another we seek simply to project the target points onto the correct side of the learned hyperplane, leading to better classification performance.

The HFA formulation also takes advantage of the max-margin framework to directly optimize the classification objective while learning transformations. HFA learns the classifier and transformations to a common latent space between the source and target. However, due to the difficulty of defining the dimension of the latent space directly, they optimize with respect to a larger combined transformation matrix and a relaxed constraint. Additionally, this transformation matrix becomes too large when the feature dimensions in source and target are large so the HFA must usually be solved in kernel space. This can make the method slow and cause it to scale poorly with the number of training examples. In contrast, our method can be efficiently solved in linear feature space which makes it fast and potentially more scalable.

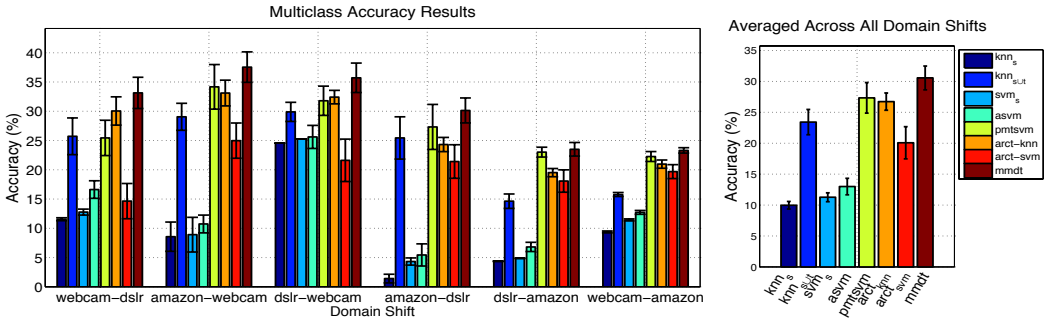

Figure 2: Multiclass accuracy evaluation on the *Office* dataset for the standard supervised domain adaptation setting where there is a small amount of training data available in target for every object category. Our method, MMDT, outperforms the no-adaptation, max-margin based, and transform based baselines.

4 Experiments on Image Datasets

We now present experiments using the *Office* [1] and *Bing* [8] datasets to evaluate our algorithm according to the following four criteria: 1) Multi-class accuracy in the standard supervised domain adaptation setting; 2) Multi-class accuracy for the supervised domain adaptation setting where the source and target have different feature dimensions. 3) Multi-class accuracy in the multi-task domain adaptation setting with novel target categories at test time; 4) Training time performance compared to the transform based ARC-t method [12].

4.1 *Office* Dataset

The *Office* dataset is a collection of images that provides three distinct domains: `amazon`, `webcam`, and `dslr`. The dataset has 31 categories consisting of common office objects such as chairs, backpacks and keyboards. The `amazon` domain contains product images (from amazon.com) containing a single object, centered, and usually on a white background. The `webcam` and `dslr` domains contain images taken in"the wild" using a webcam or a dslr camera, respectively. They are taken in an office setting and so have different lighting variation and background changes (see Figure1 for some examples.) Note that `webcam` and `dslr` contain images of the same object instances. In our experiments we follow previous protocols and evaluate on the object category recognition problem, so in the case of the `webcam` $\rightarrow$ `dslr` shift, or vice versa, we make sure different instances are used for training and testing. We use the SURF-BoW image features provided by the authors. More details on how these features were computed can be found in [1].

For our first experiment we use this domain adaptation benchmark dataset for the case when a few labeled examples are available for all categories in the target domain. This allows us to compare MMDT to the existing state-of-the-art max-margin and transform based domain adaptation methods.We follow the setup of [1]: all methods are given a training set of 20 labeled examples per category in the source domain, and 3 labeled examples per category in the target domain. In each trial run, a random subset of the eligible data is selected for training. Because each category of the office dataset may contain multiple images of the same object across domains (taken from different viewpoints, for example), only images with object IDs 1, 2, and 3 are eligible for training and only images with object IDs 4 and 5 are eligible for testing classification accuracy, ensuring that our results reflect category recognition rather than instance recognition. We use the following baselines as a comparison in the different experiments where applicable.

- **knn**$_S$: k-Nearest Neighbors using only the labeled examples in the source domain.
- **knn**$_{S\cup T}$: k-Nearest Neighbors using both the source labeled and the target labeled examples.
- **svm**$_S$: A support vector machine using only source training data.
- **asvm**: A category specific parameter adaptation technique that minimizes the distance (ℓ_2) from the target and source parameter vectors. We train an 1-vs-all A-SVM classifier [10, 9].

	svm$_s$	**svm**$_t$	**arct** [?]	**hfa** [?]	**gfk**[?]	**mmdt** (ours)
a → w	33.9 ± 3.1	62.4 ± 4.0	41.3 ± 3.5	57.5 ± 4.8	58.6 ± 4.4	64.6 ± 5.2
a → d	35.0 ± 3.5	55.9 ± 3.4	38.2 ± 3.2	52.7 ± 4.2	50.7 ± 3.6	56.7 ± 5.6
w → a	35.7 ± 2.0	45.6 ± 3.0	39.6 ± 1.7	40.2 ± 3.0	44.1 ± 1.7	47.7 ± 4.1
w → d	66.6 ± 3.3	55.1 ± 3.6	69.6 ± 4.5	52.4 ± 5.3	70.5 ± 3.3	67.0 ± 4.9
d → a	34.0 ± 1.3	45.7 ± 3.9	38.5 ± 2.0	40.1 ± 4.6	45.7 ± 2.8	46.9 ± 4.6
d → w	74.3 ± 2.4	62.1 ± 3.4	76.3 ± 2.5	58.1 ± 4.1	76.5 ± 2.3	74.1 ± 3.8
a → c	35.1 ± 1.2	32.0 ± 3.4	32.7 ± 1.2	29.8 ± 3.2	36.0 ± 2.2	36.4 ± 3.4
w → c	31.3 ± 1.6	30.4 ± 3.2	31.3 ± 2.2	27.0 ± 3.6	31.1 ± 2.8	32.2 ± 3.7
d → c	31.4 ± 1.3	31.7 ± 2.6	32.1 ± 1.3	29.0 ± 2.4	32.9 ± 2.2	34.1 ± 3.5
c → a	35.9 ± 1.8	45.3 ± 4.0	39.2 ± 2.0	39.4 ± 4.1	44.7 ± 3.5	49.4 ± 3.8
c → w	30.8 ± 5.0	60.3 ± 4.5	43.1 ± 5.2	55.9 ± 5.1	63.7 ± 3.6	63.8 ± 4.8
c → d	35.6 ± 3.3	55.8 ± 4.0	42.2 ± 3.5	52.5 ± 5.1	57.7 ± 4.8	56.5 ± 3.9
mean	40.0 ± 2.5	48.5 ± 3.6	43.7 ± 2.7	44.6 ± 4.1	51.0 ± 3.1	52.5 ± 4.3

Table 1: 10 Category case: All results are from our implementation. In the case of **gfk**, the previously published results vary slightly from our implementation. However, when averaged across all domain shifts the reported average value for the method was 51.65 while our implementation had an average of 51.0 ± 3.1. Therefore, the result difference is well within the standard deviation over data splits. Red indicates the best result for each domain split. Blue indicates the group of results that are close to the best performing result.

Table 2: Multiclass accuracy results on the standard supervised domain adaptation task with different feature dimensions in the source and target. The target domain is `dslr` for both cases.

source	**arc-t (knn)**	**arc-t (svm)**	**hfa**	**mmdt**
`amazon`	47.9 ± 2.4	51.7 ± 2.1	43.1 ± 2.7	**52.9 ± 2.8**
`webcam`	49.1 ± 2.1	50.4 ± 2.9	46.7 ± 4.0	**55.7 ± 1.9**

- **pmt-svm**: A category specific parameter adaptation technique that minimizes the angular distance between the target and source parameter vectors. We train an 1-vs-all PMT-SVM classifier [11].
- **arc-t**: A category general feature transform method proposed by [12]. We implement the transform learning and then apply both a KNN classifier (as originally proposed) and an SVM classifier.

Figure 2 shows the multiclass accuracy (%) over the test data from the target domain from each of the six possible domain shifts. On the x-axis each result cluster represents a distinct domain shift where the source and target domains are formatted as: [source]-[target].

Our method outperforms all other methods for each domain shift in this setting. The most benefit is offered for the shift of `webcam`→ `dslr` and `dslr`→`webcam`. This aligns with our intuition that if the nature of the domain shift is consistent across all object categories (in this case, the camera sensor), then learning a joint transformation by pooling all classes will perform better than learning per-class offsets (asvm and pmt-svm). Additionally, in general MMDT provides a significant improvement over the other global transformation method, arc-t, which uses similarity based constraints for learning a transformation, rather than optimizing the classification error directly.

Next, we analyze the effectiveness of our asymmetric transform learning by experimenting with the source and target having different feature dimensions. We use the same experimental setup as previously, but let the source domains (`amazon` or `webcam`) be represented with SURF features quantized into an 800-dimensional histogram feature. The target (`dslr`), however, is quantized into a 600-dimensional histogram feature.

For this asymmetric transfer setting, we can not directly compare against **knn**$_S$**, knn**$_{S\cup T}$**, svm**$_S$**, asvm**, and **pmt-svm**. We instead add an additional baseline of **hfa** [**?**], which is a max-margin transform approach that seeks to learn a latent common space between source and target as well as

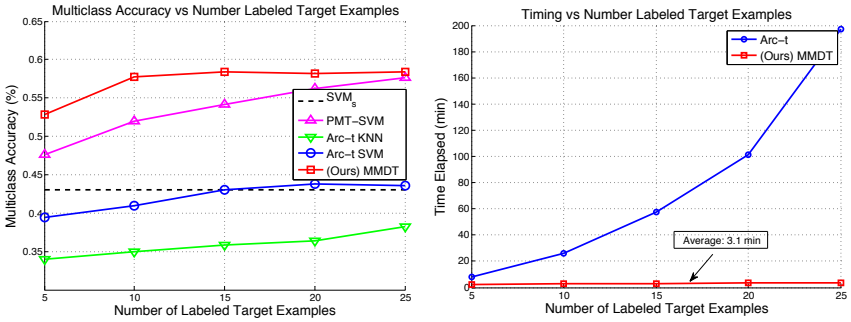

Figure 3: Left: multiclass accuracy on the *Bing* dataset using 50 training examples in the source and varying the number of available labeled examples in the target. Right: training time comparison.

Table 3: Multiclass accuracy results on the *Office* dataset for the domain shift of `webcam`→`dslr` for target test categories not seen in at training time. Following the experimental setup of [12]. We compare against pmt-svm [11] and ARC-t [12] using both knn and svm classification.

No Adaptation Baselines			Adaptation Baselines			Ours
knn$_s$	**knn**$_{s\cup t}$	**svm**$_s$	**pmt-svm**	**arc-t** (knn)	**arc-t**(svm)	**mmdt**
34.8	12.0 ± 0.6	48.1 ± 0.3	48.1 ± 0.3	44.5 ± 0.4	35.8 ± 0.8	$\mathbf{55.0 \pm 1.3}$

a classifier that can be applied to points in that common space. Table 2 shows multi-class accuracy results for this setting.[1].

An important observation is that our linear method outperforms **hfa** and **arc-t** even though they both learn a non-linear transformation using a Gaussian RBF kernel ($\sigma = 1$).

4.2 *Bing* Dataset

In the previous experiment we showed that our max-margin transform method produces higher multiclass accuracy then the baseline for the standard domain adaptation task where a small amount of labeled target data is available for every category.

With the next experiment we show that while our method gains accuracy performance it is also considerably faster than the baseline transform learning method, ARC-t [12]. As described in Section 3, the number of constraints for our optimization problem scales linearly with the number of labeled target points, n_T, and does not depend on the number of source labeled points. Conversely, the number of constraints that need to be optimized for the ARC-t baseline is equal to the product of the number of labeled points in the source and the target, $n_S \cdot n_T$.

To demonstrate the effect that constraint set size has on run-time performance, we use the *Bing* dataset from [8], which has a larger number of images in each domain than *Office*. The source domain has images from the Bing search engine and the target domain is from the *Caltech256* benchmark. We run experiments using the first 20 categories and set the number of source examples per category to be 50. We use the train/test split from [8] and then vary the number of labeled target examples available from 5 to 25. The left-hand plot in Figure 3 presents multiclass accuracy for this setup. Our MMDT method provides a considerable improvement over the ARC-t method in terms of multiclass accuracy, and beats PMT-SVM for smaller numbers of target examples, again confirming the benefit of multi-task learning. Additionally, the training time of our method (run to convergence) and that of ARC-t (just transform-learning) is shown empirically on the right-hand plot in Figure 3. MMDT takes on average 3.8 seconds to learn a feature transformation in step(1) (3.1 minutes to run to convergence) while ARC-t takes up to 200 minutes for the largest experiment.[2] Consider the scenario with 50 examples per category in the source ($n_S = 50 * K$) and 20 training examples per category in the target ($n_T = 20 * K$). For 20 categories, MMDT needs to optimize $K \cdot n_T = (20)(20 * 20) = 8000$ constraints. Conversely, the ARC-t baseline method needs to

[1]Note that for this experiment we used both the code and the parameter settings reported by [**?**] for the **hfa** method, but our result had a lower accuracy than reported in [**?**]

[2]We used the LIBSVM package [**?**]; faster linear SVM solvers could further speed up our method.

optimize $n_S \cdot n_T = (20 * 50) * (20 * 20) = 400,000$ constraints. Therefore, this experiment demonstrates that when the number of source data points is large, the ARC-t method does not scale well, while our MMDT method is able to learn a transformation on a large source dataset very quickly.

4.3 Generalizing to Novel Categories

Finally, we consider the setting of practical importance where labeled target examples are not available for all objects. Recall that this is a setting that many category specific adaptation methods cannot generalize to. Therefore, we compare our results for this setting to the ARC-t [12] method which learns a category independent feature transform. The ARC-t method is presented as a technique for learning a transform to project target features into the source that can then be classified using a source classifier. Therefore, we experimented with learning the transform according to [12] and then classified the projected target features using a k-NN classifier (as was done in [12]) and additionally an SVM classifier. Specifically, we classify the baseline transformed features using the same source SVM classifier that was used as initialization to our max-margin feature transform method.

In addition we also present the PMT-SVM method as a baseline for this setting. However, since this method can not be applied directly so we first learn the individual transforms for the known categories and average them to provide a category independent shift.

Following the experimental setup of [12] we use the *Office* dataset and allow 20 labeled examples per category in the source and 10 labeled examples for the first 15 object categories in the target. The experimental results for the domain shift of `webcam`→`dslr` are evaluated and shown in Table 3; MMDT outperforms all baselines. One additional point to note is that the averaging PMT-SVM approach performs identically to the source SVM approach, indicating that averaging the offsets learned per category in PMT-SVM does not yield any category independent information.

5 Conclusion

In this paper we presented a linear SVM domain adaptation technique that combines the ability of feature transform-based methods to perform multi-task adaptation with the performance benefits of directly adapting classifier parameters. We validated the computational efficiency and effectiveness of our method using two standard benchmarks used for image domain adaptation. Our experiments show that our method is a competitive domain adaptation algorithm and is successfully able to generalize to novel target categories at test time. In addition, these benefits are offered through a framework that is both faster than prior transform-based methods and achieves higher classification accuracy.

So far we have focused on linear transforms because of its speed and scalability; however, our method can also be kernelized to include nonlinear transforms.

References

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

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

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

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

[5] A. Torralba and A. Efros. Unbiased look at dataset bias. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[6] D. McAllester P. Felzenszwalb, R. Girshick and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9), 2010.

[7] Lubomir Bourdev and Jitendra Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *International Conference on Computer Vision (ICCV)*, 2009.

[8] A. Bergamo and L. Torresani. Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In *Neural Information Processing Systems (NIPS)*, December 2010.

[9] J. Yang, R. Yan, and A. Hauptmann. Adapting svm classifiers to data with shifted distributions. In *In ICDM Workshops*, 2007.

[10] X. Li. Regularized adaptation: Theory, algorithms and applications. In *PhD thesis, University of Washington, USA*, 2007.

[11] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2011.

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

[13] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *13th International Conference on Computer Vision 2011*, November 2011.

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

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

[16] H. Daume III. Frustratingly easy domain adaptation. In *ACL*, 2007.

[17] S. Ben-david, J. Blitzer, K. Crammer, and O. Pereira. Analysis of representations for domain adaptation. In *In NIPS*. MIT Press, 2007.

[18] J. Jiang and C. X. Zhai. Instance weighting for domain adaptation in nlp. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[19] J. Jiang. A literature survey on domain adaptation of statistical classifiers. `http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/`.

[20] W. Jiang, E. Zavesky, S. Chang, and A. Loui. Cross-domain learning methods for high-level visual concept classification. In *ICIP*, 2008.