Continuous Manifold Based Adaptation for Evolving Visual Domains

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

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

Kate Saenko
UMass Lowell, CS
saenko@cs.uml.edu

Abstract

We pose the following question: what happens when test data not only differs from training data, but differs from it in a continually evolving way? The classic domain adaptation paradigm considers the world to be separated into stationary domains with clear boundaries between them. However, in many real-world applications, examples cannot be naturally separated into discrete domains, but arise from a continuously evolving underlying process. Examples include video with gradually changing lighting and spam email with evolving spammer tactics. We formulate a novel problem of adapting to such continuous domains, and present a solution based on smoothly varying embeddings. Recent work has shown the utility of considering discrete visual domains as fixed points embedded in a manifold of lower-dimensional subspaces. Adaptation can be achieved via transforms or kernels learned between such stationary source and target subspaces. We propose a method to consider non-stationary domains, which we refer to as Continuous Manifold Adaptation (CMA). We treat each target sample as potentially being drawn from a different subspace on the domain manifold, and present a novel technique for continuous transform-based adaptation. Our approach can learn to distinguish categories using training data collected at some point in the past, and continue to update its model of the categories for some time into the future, without receiving any additional labels. Experiments on two visual datasets demonstrate the value of our approach for several popular feature representations.

1. Introduction

It has become increasingly clear that there is a significant bias between available labeled visual training data and the data encountered in the real world [28, 16]. Unfortunately, supervised classifiers trained on one distribution often fail when faced with a different distribution at test time. Domain adaptation techniques offer a way to transfer information learned from source (training) data to the eventual target (test) domain, so as to diminish the performance degradation and "learn from the past." Supervised adaptation methods assume a few labeled target examples are available [17, 5, 12]. However, obtaining these is often expensive or impossible, so unsupervised adaptation is of particular importance [10, 9, 7].

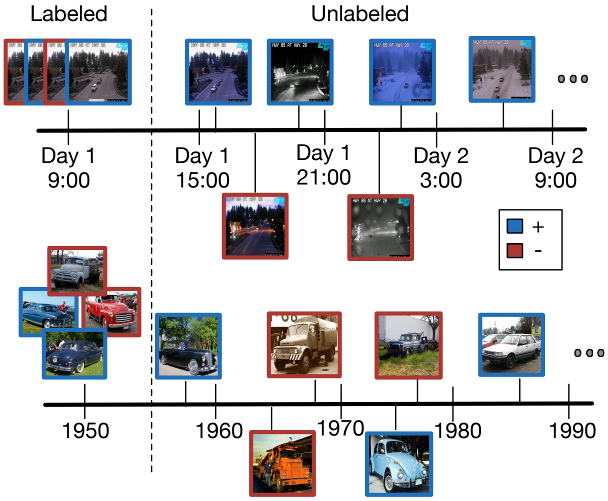

Figure 1. Problem setup: We want to classify test data drawn from an evolving distribution (target domain), using labeled data from a distribution collected in the past (source domain). We show two example scenarios. ABOVE: classifying traffic scenes streaming from a traffic camera as busy (blue border) or empty (red border). BELOW: classifying sedans (blue border) vs trucks (red border) across many decades as the design and shapes of the two evolve.

In this paper, we address the problem of unsupervised adaptation to a *continuously evolving* target distribution. Specifically, we assume that

1. ample labeled data is available in the source domain,
2. the target domain examples are unlabeled and arrive sequentially,
3. the target distribution evolves over time.

One scenario where this problem occurs is object or scene classification in video streams. For example, classifying scene types in a video feed from a traffic camera is challenging. The appearance of the same scene type (class) in the target domain is constantly changing due to sunlight/shadows, time of day, sensor change to IR at nighttime, and unexpected weather patterns. Another example is classifying objects or scenes as their appearance evolves over time. These two problems are illustrated in Figure 1. Also, while we focus on visual tasks in this paper, the problem also occurs in spam filtering, where spammers constantly change their tactics to deceive email users, and sentiment analysis in social media. Current unsupervised domain adaptation methods cannot naturally handle such problems. They assume that the target distribution is stationary, and that a large amount of unlabeled data is available in batch for modeling this fixed target distribution.

We stress that traditional online learning methods are not suitable for our problem. Online learning methods for classification use sequentially arriving data, but require that data to be labeled. In contrast, online distribution learning can be used for estimating an evolving domain [20, 23] , but provides no means for adapting a classifier between domains which makes it insufficient for our task. We found that learning an evolving distribution without adaptation had worse performance than classifying in the original feature space. Finally, online adaptation methods do learn from streaming observations without labels [4, 8], but expect to learn a single, stationary target distribution.

We propose a novel adaptation method which models continuously changing domain distributions by forming incremental, sample-dependent adaptive kernels. Our approach is inspired by recent methods that learn a transformation in feature space to minimize domain-induced dissimilarity [24, 17, 5, 12]. In unsupervised adaptation, this can be accomplished by projecting all source and target data points to their respective lower dimensional subspaces, and then minimizing the distance between the subspaces to compute a domain-invariant kernel [10, 9, 7].

However, a major limitation of these methods is that all target points are assumed to belong to a single target domain, or split into several domains with known boundaries. To apply them in our scenario, we must discretize the evolving target domain into a set of fixed domains. For the traffic camera example in Figure 1, this would treat all of the changes within a certain time window as a single target domain. This is problematic, as it may apply the same adaptation to, say, sunny conditions, snow storm, and night time images. A second major limitation with these methods is that data is expected in batch.

We argue that it is more natural to model the domain shift in a continuously adaptive fashion. Our technique works by learning the optimal lower dimension subspace for a specific test sample, rather than embedding it in a single monolithic subspace encompassing all of the unlabeled target data. A key advantage of our method is that there is no need to segment the test samples into a discrete set of domains, either manually or automatically, and thus no need to model the number or size of such domains. Another important advantage of our approach is the ability to more precisely adapt to each test example in an online fashion. This is helpful in situations when test samples are not available in batch but arrive sequentially. While we present an unsupervised approach, the ideas can be applied to supervised scenarios as well.

2. Related Work

Domain adaptation has been extensively studied in speech recognition, natural language processing and machine learning. More recently, domain adaptation techniques have been applied to visual datasets. Several supervised parameter-based adaptation methods have been proposed to learn a target classifier with a small amount of labeled training data, by regularizing the learning of a new parameter against an already learned source classifier [29, 1]. Other supervised methods learn feature transformations between source and target distributions, so classifiers may be trained directly in the source and applied to transformed target points, or trained on transformed source and transformed target data jointly [24, 17]. Some methods seek to benefit from both the discriminative power of parameter-based approaches and the flexibility of the feature-transformation approaches through unified optimization frameworks [5, 12].

A recent class of unsupervised domain adaptation techniques attempts to align the unlabeled target data with the source using manifold learning. Domains are represented as subspaces embedded in a Grassman manifold, and adaptation is carried out through geodesic flow computations on this manifold [10, 9, 7]. However, none of these methods has considered our setting of non-discrete, continuously evolving domains. They also require all unlabeled target data to be available in batch and are not designed for online adaptation. [11] argued that datasets are composed of multiple hidden domains, which they estimate via constrained clustering, however, the number of domains is discrete and no online solution is proposed.

Supervised online learning allows a classifier to be trained with sequentially arriving data. At each round the learner receives a data point, and predicts its label. The correct answer is then revealed and the learner suffers a loss [25]. Such methods can be used to "adapt" to the incoming data stream by controlling the learning rate. In our setting, however, labels exist only in the source domain, and supervised online learning cannot be carried out.

In vision, a classic online adaptive approach is background subtraction (see [2] for a review), where the distribution of pixels belonging to the background is continuously updated. However, in classification, we are interested in categorizing the entire scene (or object), not distinguishing between foregrond and background (although we can do that as a preprocessing step). In detection, a method for online adaptation was proposed that bootstraps offline classifiers to obtain new labels and uses them to continually update car detectors in a traffic scene [15, 13]. However detection fails on our traffic camera task due to the extremely low resolution of individual objects.

The natural language processing and speech recognition communities have developed algorithms to tackle the task of online adaptation. In speech recognition, the recognition of a new speaker's speech can and should be adapted and improved over time. Online incremental unsupervised fMLLR [8] dynamically collects acoustic statistics from the speaker and updates the acoustic models. [4] combines parameters of multiple classifiers to do online adaptation of spam classifiers for individual users, as well as sentiment prediction for books, movies and appliances. However, the domain change due to a new speaker or a new email user is discrete. While examples may arrive sequentially, they all arise from the same distribution (the same speaker, or the same user). On the other hand, our approach "tracks" the evolving distribution, and in that way it is somewhat akin to distribution tracking methods common in the signal processing literature, e.g., Kalman filtering [14], [21].

3. Approach

3.1. Background: Unsupervised Adaptation Using the Data Manifold

We build on a class of methods recently proposed for unsupervised adaptation [10, 9, 7], which are based on modeling the data manifold. Their key insight is that visual data have an inherent low dimensional structure, and can thus be embedded in lower dimensional subspaces. Furthermore, these subspaces lie on a Grassman manifold of the same dimension. By exploiting the properties of the manifold, such as smoothness, we can find a novel embedding that compensates for the differences between domains.

Suppose we have a set of labeled examples drawn from a source domain, $x_1, \ldots, x_{n_S} \in \mathbb{R}^D$, with labels $y_1, \ldots, y_{n_S}$. At test time, we receive unlabeled examples drawn from the target domain, $z_1, \ldots, z_{n_T} \in \mathbb{R}^D$, which are distributed differently from the source examples. For now we assume that the target examples come from a single, stationary distribution; we will relax this assumption shortly.

To account for discrepancies between the training (source) and test (target) distributions, we seek to learn a linear transformation W that maps source points in a way that makes their distribution more similar to that of the target points. Such a transformation can then be applied to compute a kernel $x^T W z$, which can be used in any inner-product based classifier. An alternative is to factor the transformation into two transformations $W = AB^T$, where A and B embed source and target points, respectively, in a new subspace.

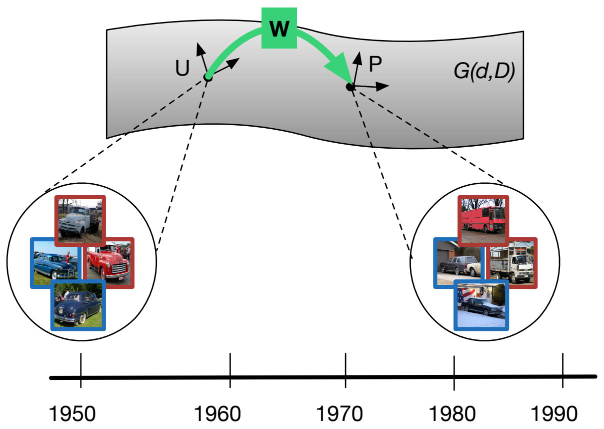

Figure 2. Conventional adaptation techniques separate samples into a discrete set of domains, seen here as points on a domain manifold (a single source domain S and target domain T).

To find W, we assume the source and target domains lie on lower dimensional orthonormal subspaces, $\boldsymbol{U}, \boldsymbol{P} \in \mathbb{R}^{D \times d}$, which are points on the Grassman manifold, $\mathcal{G}(d, D)$ (See Fig. 2), where $d \ll D$. Several techniques exist for finding such low dimensional embeddings, including Principal Component Analysis. We then reformulate our goal as finding embeddings $\tilde{A}$ and $\tilde{B}$ that map the low dimensional subspaces in such a way as to make them better aligned. This objective can be formalized as minimizing the distance between the two projected subspaces, $\boldsymbol{U}\tilde{A}$ and $\boldsymbol{P}\tilde{B}$.

$$\min_{\tilde{A},\tilde{B}} \qquad \psi(\boldsymbol{U}\tilde{A}, \boldsymbol{P}\tilde{B}) \tag{1}$$

One recent approach, called the Subspace Alignment (SA) method [7], solves the unconstrained optimization problem in Equation (1) directly, setting the subspace distance metric to be the Frobenius norm difference: $\psi(\boldsymbol{U}\tilde{A}, \boldsymbol{P}\tilde{B}) = \|\boldsymbol{U}\tilde{A} - \boldsymbol{P}\tilde{B}\|_F^2$. Since both $\boldsymbol{U}$ and $\boldsymbol{P}$ are orthonormal matrices, the global minimizer for this subspace distance metric is reached when $\tilde{A} = \boldsymbol{U}^T\boldsymbol{P}$ and $\tilde{B} = I$. This leads to the following transformation between points in the original spaces: $W_{\text{SA}} = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{P}\boldsymbol{P}^T$.

Another recent method that seeks to find embeddings for the source and target points, so as to minimize the distance between their distributions, is the Geodesic Flow Kernel (GFK) [9]. This method learns a symmetric embedding ($A = B$) by computing the geodesic flow along the manifold, $\boldsymbol{\phi}(\cdot)$. The flow is constructed in such a way that it starts at the source subspace at time 0, $\boldsymbol{U} = \boldsymbol{\phi}(0)$, then reaches the target subspace in unit time: $\boldsymbol{P} = \boldsymbol{\phi}(1)$. The intuition is to project all source and target points into all intermediate subspaces along the flow between the source

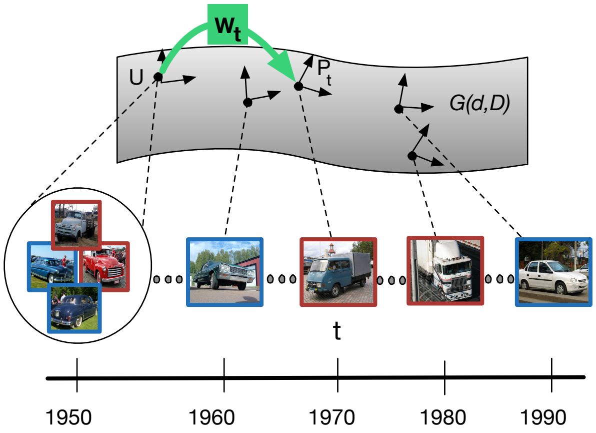

Figure 3. Our approach (CMA) treats each target sample as arising from a different point (ex: indexed by time) along the continuous domain manifold, resulting in more precise adaptation.

and target subspace. The final transformation is then computed by integrating over the infinite set of all such intermediate subspaces between the source and target $W_{\text{GFK}} = \int_0^1 \boldsymbol{\phi}(\ell)\boldsymbol{\phi}(\ell)^T d\ell$, which has a closed form solution presented in [22, 9].

3.2. Adapting to Continuously Evolving Domains

We seek to adapt to and classify streaming target data that is drawn from a continuously evolving distribution. The drawback of the above methods is that they require discrete known domains, where the data from each domain is available in batch (see Figure 2). To adapt to each instance the above methods would need to artificially discretize the target by using a fixed windowed history and would still fail to adapt until enough data had arrived to begin learning subspaces. This is not what the method was originally designed for, would be very computationally expensive and would require cross-validating or tuning a hyperparameter to choose the appropriate window size. Next, we present our approach, Continuous Manifold Adaptation (CMA), which does not require knowledge of discrete domains (see Figure 3).

Suppose that at test time, we receive a stream of observations $z_1, \ldots, z_{nT} \in \mathbb{R}^D$, which arrive one at a time, and are drawn from a continuously changing domain.[1] We assume the distribution of possible points arriving at t can be represented by a lower dimensional subspace $\boldsymbol{P}_t$.

To align the training and test data, we seek to learn a time-varying transformation, W_t, between source and target points, where t indexes the order in which the examples are received. As presented in Section 3.1, this transformation can equivalently be written as learning two time-varying embeddings that map between points of the two lower dimensional subspaces, $\tilde{A}_t$ and $\tilde{B}_t$, with the mapping in the original space being defined as $W_t = \tilde{A}^T \boldsymbol{U}^T \boldsymbol{P}_t \tilde{B}_t$. This computes a time varying kernel between the source data and the evolving target data $x^T W_t z_t$ which can be used with any inner product based classifier.

Since we no longer have a fixed target distribution with all examples delivered in batch, we must simultaneously learn the lower dimensional subspace, $\boldsymbol{P}_t$, representing the distribution from which the data was drawn at each time t. We will search for a subspace that minimizes the reprojection error of the data:

$$R_{err}(z_t, \boldsymbol{P}_t) \quad = \quad \|z_t - \boldsymbol{P}_t(\boldsymbol{P}_t^T z_t)\|_F^2 \qquad (2)$$

In general, we may receive as few as one data point at each time step so we will regularize our subspace learning by a smoothness assumption that the target subspace does not change quickly.[2]

Therefore, at each time step, our goals can be summarized by optimizing the following problem:

$$\min_{\boldsymbol{P}_t^T \boldsymbol{P}_t = I, \tilde{A}_t, \tilde{B}_t} r(\boldsymbol{P}_{t-1}, \boldsymbol{P}_t) + R_{err}(z_t, \boldsymbol{P}_t) + \psi(\boldsymbol{U}\tilde{A}_t, \boldsymbol{P}_t \tilde{B}_t) \qquad (3)$$

where $r(\cdot)$ is a regularizer that encourages the new subspace learned at time t to be close to the previous subspace of time $t-1$.

Equation (3) is a non-convex problem and we choose to solve it by alternating between the three steps below:

1. Receive data z_t
2. Given $\tilde{A}_{t-1}$ and $\tilde{B}_{t-1}$ compute $\boldsymbol{P}_t$
3. Given $\boldsymbol{P}_t$ compute $\tilde{A}_t$ and $\tilde{B}_t$

To optimize step 2, we begin by fixing $\tilde{A}_{t-1}$ and $\tilde{B}_{t-1}$ and then we examine the third term of the optimization function. Note that it would be minimized if $\boldsymbol{P}_t = \boldsymbol{P}_{t-1}$. Therefore, with a fixed $\tilde{A}_{t-1}$ $\tilde{B}_{t-1}$, the term $\psi(\boldsymbol{U}\tilde{A}_{t-1}, \boldsymbol{P}_t \tilde{B}_{t-1})$ is acting as a regularizer that penalizes when $\boldsymbol{P}_t$ deviates from $\boldsymbol{P}_{t-1}$. We therefore can equivalently solve this problem by grouping the first and third term into a single regularizer of $\boldsymbol{P}_t$ that enforces a smoothness between the subsequent learned subspaces. Finally, we can express this subproblem as follows:

$$\begin{aligned} \min_{\boldsymbol{P}_t} \quad & r(\boldsymbol{P}_{t-1}, \boldsymbol{P}_t) + R_{err}(z_t, \boldsymbol{P}_t) \\ \text{s.t} \quad & \boldsymbol{P}_t^T \boldsymbol{P}_t = I \end{aligned} \qquad (4)$$

We first observe that solving Equation (4) for the trivial regularizer $r(\cdot, \cdot) = $ constant would result in $\boldsymbol{P}_t$ which is equal to the d largest singular vectors of the data z_t,

[1] Our formulation can also be extended to the case of streaming source observations.

[2] Our model can be extended to allow for discontinuities, but we leave this as future work.

which can be obtained via SVD. Obviously, we prefer to use a non-trivial regularizer, as we don't have enough data at time t to compute a robust SVD, and also want to make sure that the subspaces vary smoothly over time. Thus we solve this optimization problem with a variant of sequential Karhunen-Loeve [20], which adapts a subspace incrementally and trades-off changing the subspace with minimizing re-projection error of z. For optimization details see [23].

When optimizing step 3, we note that the first two terms of the objective function are not active for this sub-problem, and we are left with the task of minimizing Equation (1). This is just the task of aligning two known subspaces. We experiment with solving this optimization using either of the two methods described in Section 3.1.

4. Experiments and Results

We present performance on both a scene classification experiment and an object classification experiment. For both experiments we compare our Continuous Manifold Adaptation (CMA) approach using two different unsupervised adaptation techniques (Geodesic flow kernel (GFK)[9] and Subspace Alignment method (SA)[7]) for solving Step 3 in our algorithm and two different inner product based classifiers: k-nearest neighbors (KNN with $k = 1$) and support vector machines (SVM) – trained with source data only. We demonstrate performance increase using our CMA method across a variety of feature spaces for these tasks. The unsupervised adaptation methods can not be directly applied to our problem with the streaming test domain. However, for completeness we tried learning subspaces from a fixed windowed history and then used the unsupervised adaptation approaches. We ran experiments evaluating the performance of various window sizes (including using all history available, which is computationally infeasible in practice), but were unable to find a result that was competitive with our method.

4.1. Scene Classification Over Time

Dataset Our first experiment evaluates our algorithm on scene classification over time using a real-world surveillance dataset. The images were captured from a fixed traffic camera observing an intersection. Frames were updated at 3 minute intervals each with a resolution of 320x240[3]. Our dataset includes images captured over a 2 week period. This data offers a challenging domain shift problem as changes include illumination, shadows, fog, snow, light saturation from oncoming sedans, change to night time IR mode, etc.

Experiment Setup We define an intersection traffic classification task, which is to determine whether one or more cars are present in, or approaching, the intersection. We obtain labels for this task using human annotators (for example labels see Figure 4). We assume to be given 50 labeled consecutive images (2.5 hours) and then evaluate each algorithm on the immediately following 24 hours (480 images) and 5 days (2400 images). We evaluate the classifiers in the online setting, where classification must occur just after receiving a test point and may only be informed by previously received test data with no knowledge of future test data.

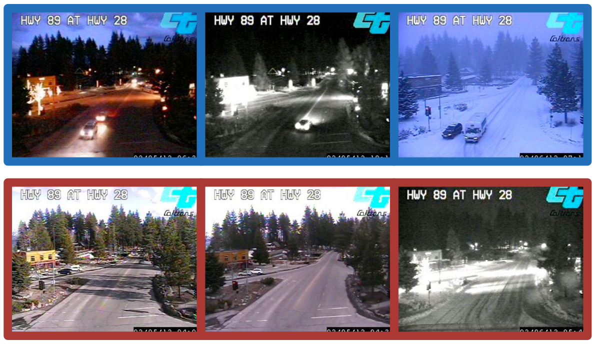

Figure 4. Sample human labeled images used for intersection traffic classification. Positive examples are shown in the top row (blue) and negative examples are shown in the bottom row (red).

This task is challenging and cannot be adequately solved with approaches such as scanning-window car detection, as the images (and especially the cars within) are too low-resolution to be detected by conventional methods. A deformable parts model (DPM) detector [6] failed to detect any sedans in the first 50 images. Instead we compute features over the whole image and produce a scene label.

We consider two features that are known to perform well on scene classification tasks: GIST [26] (512 dimension) and SIFT-SPM [18] using a 200 dimension codebook and 3 pyramid layers (4200 dimension). Finally, since the images are all of a fixed scene we use a standard mixture of gaussians background subtraction algorithm [27] to extract a foreground mask and compute the same GIST and SIFT-SPM on the foreground. We found that sequential images were far too noisy to provide useful foreground masks; we present all results here for completeness.

Results & Analysis Table 1 presents the average precision (%) when testing on the 24 hours immediately following the labeled data. CMA is shown to provide improvement over no-adaptation regardless of feature choice. The strongest algorithm and feature combination for this setup was to use CMA with GIST features and either type of subspace alignment algorithm and either classifier.

We next demonstrate that the algorithm does not diverge and in fact continues to provide improvement by testing over the a 5 day period (see Table 2). Here we show results using the GIST feature with each type of classifier and adap-

[3] Available at the California Department of Transportation website, http://quickmap.dot.ca.gov/

Adaptation Method	Classifier	GIST[26]	SIFT-SPM[18]	GIST[26] + BSub[27]	SIFT-SPM[18]+BSub[27]
-	KNN	76.30± 3.0	47.51±5.1	52.27±3.4	39.91±3.0
-	SVM	74.42± 3.0	68.69±3.6	50.98±3.6	48.91±3.0
CMA+GFK	KNN	**78.07**±**1.8**	49.84±5.5	52.97±2.7	39.08±2.6
CMA+GFK	SVM	**78.38**± **3.1**	74.98±2.7	59.55±2.9	47.59±2.8
CMA+SA	KNN	**78.71**±**1.7**	54.08±6.2	51.33±4.2	38.21±2.6
CMA+SA	SVM	**78.49**±**3.1**	75.66±2.9	59.68±2.9	49.05±2.8

Table 1. Our method, CMA, improves performance independent of the feature choice for the scene classification task. Results here are shown with optimizing the unsupervised adaptation problem using either the geodesic flow kernel (GFK)[9] or the subspace alignment (SA) method [7]. Average precision (%) is recorded when training with 50 labeled images and testing on the immediately following 24 hours (480 images).

Adaptation	Classifier	GIST[26]
-	KNN	71.24±5.7
-	SVM	80.40±0.6
CMA+GFK	KNN	77.21± 3.8
CMA+GFK	SVM	**84.17**±**1.5**
CMA+SA	KNN	78.61±3.3
CMA+SA	SVM	**84.32**±**1.4**

Table 2. Our method, CMA, continues to provide improvement for the scene classification task even when testing over the 5 days following the labeled training data. We show here average precision (%) for the 2400 test images following the 50 available labeled training images.

tation optimization algorithm. We found that SVM generalized better over time.

To understand the performance of the adaptive method, we examine qualitative classification examples. Figure 5 shows images that were misclassified by all algorithms except our CMA approach. The sedans parked in the parking lot on the left side of the image as well as the protrusion from the snow mound between the road and turn-out were likely confusions for the non-adaptive baselines. Figure 6 shows images incorrectly classified by all algorithms. Here are negative examples that may have sedans present, but too far away to be considered traffic at the intersection by our task definition.

For reference, if one had access to all of the test data in **batch** one could directly apply an adaptation methods or even pre-cluster the test data and learn multiple transformations. The performance for batch test data using GIST features with SA and SVM is 76.44 AP for the single cluster case and 77.57 AP for the multiple cluster case. These are both for the 1 day test set. We see here that actually our algorithm is performing even better than using the algorithms in batch with pre-clustering of the data.

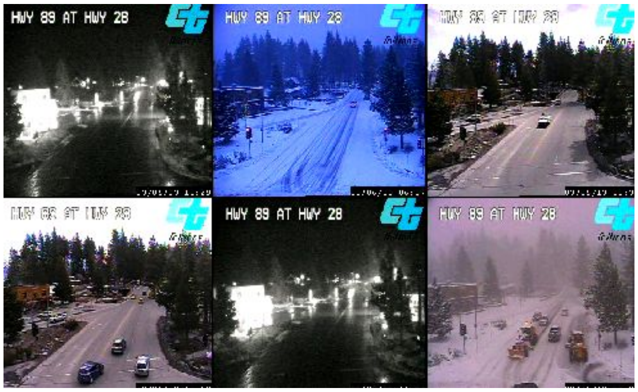

Figure 5. Qualitative results from the intersection traffic classification task. Training on day-time images with no snow only. Images labeled correctly by our method (CMA) and incorrectly labeled by all other methods. We show here the 6 examples for which the baseline had highest (incorrect) confidence, indicating that these examples were particularly challenging for the baseline and then fixed with our method. We improve in the cases of nighttime, snow, and fog, not seen during training.

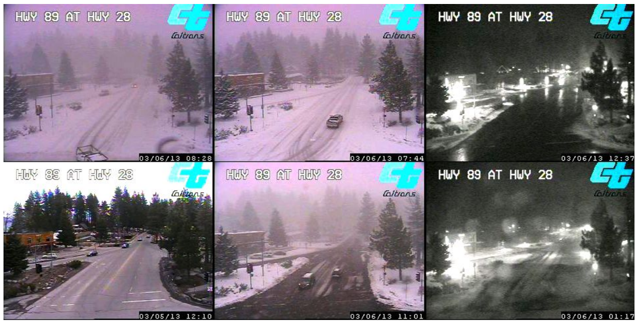

Figure 6. Qualitative results from the intersection traffic classification task. Example images where all methods classified incorrectly – snow, sedans too far away, and bright lights in the distance make these images difficult.

Adaptation	Classifier	SIFT-SPM [18]	GIST [26]	DeCAF [3]
-	KNN	66.31± 0.6	72.77± 0.8	84.60± 0.7
-	SVM	79.26± 0.6	76.40± 0.7	85.92± 0.4
CMA+GFK	KNN	66.32± 0.2	72.60± 0.9	82.65± 0.5
CMA+GFK	SVM	80.24± 0.7	78.32± 0.6	**89.68± 0.1**
CMA+SA	KNN	65.06± 1.1	71.44± 1.3	81.97± 0.6
CMA+SA	SVM	79.79± 0.6	78.31± 0.7	**89.71± 0.1**

Table 3. Our algorithm improves performance on category recognition task. We evaluate our continuous manifold adaptation approach (CMA) on the task of labeling images of automobiles as either cars or trucks. We show results using two solutions to the unsupervised adaptation problem (GFK[9] and SA[7]) and two inner product based source classifiers (KNN and SVM). We compare across three types of features and demonstrate the benefit of using our algorithm for each feature choice, including a deep learning based feature that was tuned for object classification on all of ImageNet[3].

4.2. Object Classification Over Time

Dataset Next, we evaluate on the task of distinguishing sedans and trucks over time. We collected a new automobile dataset that contains images of automobiles manufactured between the years of 1950-2000. The data was acquired from a freely available online database[4] that has object centric images of automobiles, each user labeled with a manufactured year and a model label. This database was recently proposed for detecting connections in space and time [19]. The images vary in size but are usually around 400x600. We collected 30-40 images (depending on availability) from each year of the images that were tagged as either a sedan or a truck. We directly used those tag labels as our ground truth for the car and truck classes. See Figure 1 (bottom row) for example images.

Experiment Setup Our task is to classify each test image as either a car or a truck. We use the first 5 years of data (1950-1954) as our labeled source examples. We then consider receiving all subsequent test data sequentially in time. As in the previous experiment we use both GIST [26] (512 dimension) and SIFT-SPM [18] using a 200 dimension codebook and 3 pyramid layers (4200 dimension) representations for this data. Additionally, as this is an object classification task, we also experiment with a recently proposed feature based on vectorizing a layer of a deep learning architecture trained on all of ImageNet, called DeCAF [3].[5]

Results & Analysis We present classification accuracy results on the automobile dataset in Table 3. All results represent an average across 10 random train/test splits. Our algorithm, CMA, provides a significant accuracy improvement over the non-adaptive baselines for the GIST and DeCAF features. The best overall results, with 90% accuracy, were achieved using the DeCAF features and our CMA approach followed by an SVM classifier.

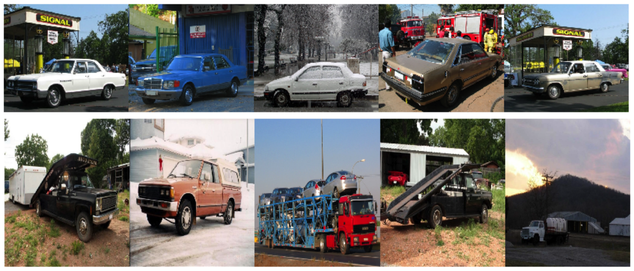

Figure 7. Our method clearly adapts to vehicle appearance as it evolves to look different from that in the labeled 50's training data. We show example images misclassified by non-adaptive SVM (DeCAF features) and correctly classified by CMA followed by the same SVM classifier. The 5 sedans and 5 trucks for which the SVM had the highest confidence (though incorrect) are displayed here.

To get a sense for which examples CMA provides improvement, we looked at the set of images that were incorrectly classified by a non-adaptive source SVM and then were correctly classified by CMA. We then displayed the 5 car and 5 truck examples for which the SVM has the highest (incorrect) confidence – indicating these were difficult examples (see Figure 7). In particular, they include sedans on top of trucks and trucks with ramps off the back.

There were also examples for which all methods misclassified the results (see Figure 8). All algorithms were consistently confused by vans and pickup trucks with covered beds, labeling them as sedans (though it's debatable which category the vans should fall into anyway). Additionally, sedans with distinctive front grates or high profiles sedans were sometimes confused with trucks. There were in general more mislabeled trucks than sedans.

5. Conclusion

We have presented a novel problem statement of performing a visual classification task under dynamic distribution shift. Our solution method dynamically learns data specific subspaces through time in order to compute an adaptive transformation at each time step. We experimentally validate that our algorithm outperforms non-adaptive baselines, independent of feature representation, and across two real world visual adaptation tasks where the target is dynamically distributed over time.

In this paper, we focused on the unsupervised learning task because of its practical importance, but in the future we would like to examine the performance benefit of adding

[4] http://www.cardatabase.net

[5] For our experiments we use the vectorized output of layer 6 of the network.

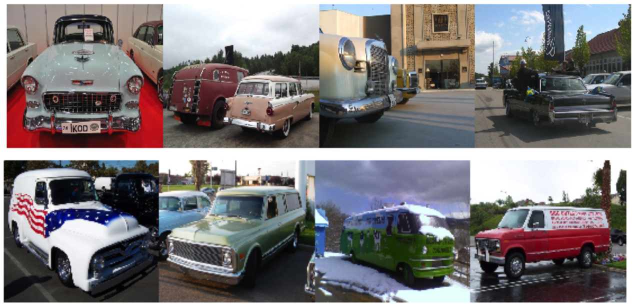

Figure 8. Example images misclassified by all methods (sedans top and trucks bottom). Vans and trucks with covered beds were consistently labeled as sedans by all algorithms. Additionally, sedans with distinctive front grates and/or high profiles were sometimes confused with trucks.

a few labeled target examples in an active learning framework. We suspect that especially in the setting where there are sudden dramatic shifts in the data, the discrepancy is perceptible by the algorithm and some supervision could focus the subspace learning and boost performance.

Acknowledgement This research was supported in part by DARPA Mind's Eye and MSEE programs, by NSF awards IIS-0905647, IIS-1134072, and IIS-1212798, and by support from Toyota.

References

[1] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *Proc. ICCV*, 2011.

[2] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger. Review and evaluation of commonly-implemented background subtraction algorithms. In *Proc. ICPR*, 2008.

[3] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. *ArXiv e-prints*, 2013.

[4] M. Dredze and K. Crammer. Online methods for multi-domain learning and adaptation. In *Proc. EMNLP*, 2008.

[5] L. Duan, D. Xu, and Ivor W. Tsang. Learning with augmented features for heterogeneous domain adaptation. In *Proc. ICML*, 2012.

[6] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2010.

[7] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Proc. ICCV*, 2013.

[8] D. Giuliani, R. Gretter, and F. Brugnara. On-line speaker adaptation on telephony speech data with adaptively trained acoustic models. In *Proc. ICASSP*, 2009.

[9] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Proc. CVPR*, 2012.

[10] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Proc. ICCV*, 2011.

[11] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko. Discovering latent domains for multisource domain adaptation. In *Proc. ECCV*, 2012.

[12] J. Hoffman, E. Rodner, J. Donahue, K. Saenko, and T. Darrell. Efficient learning of domain-invariant image representations. In *Proc. ICLR*, 2013.

[13] V. Jain and E. Learned-Miller. Online domain adaptation of a pretrained cascade of classifiers. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 2011.

[14] R Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 1960.

[15] A. Kembhavi, B. Siddiquie, Roland Miezianko, Scott McCloskey, and L.S. Davis. Incremental multiple kernel learning for object recognition. In *Computer Vision, 2009 IEEE 12th International Conference on*, 2009.

[16] A. Khosla, T. Zhou, T. Malisiewicz, A. Efros, and A. Torralba. Undoing the damage of dataset bias. In *Proceedings of the 12th European conference on Computer Vision*, 2012.

[17] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[18] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition (CVPR)*, 2006.

[19] Y. J. Lee, A. Efros, and M. Hebert. Style-aware mid-level representation for discovering visual connections in space and time. In *Proc. ICCV*, 2013.

[20] A. Levey and gM. Lindenbaum. Sequential karhunen-loeve basis extraction and its application to images. *Image Processing, IEEE Transactions on*, 2000.

[21] X. Li, K. Wang, W. Wang, and Y. Li. A multiple object tracking method using kalman filter. In *Information and Automation (ICIA), 2010 IEEE International Conference on*, 2010.

[22] Q. Rentmeesters, P-A Absil, P. Van Dooren, K. Gallivan, and A. Srivastava. An efficient particle filtering technique on the grassmann manifold. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, 2010.

[23] D. Ross, J. Lim, and M.H. Yang. Adaptive probabilistic visual tracking with incremental subspace update. In *European Conference on Computer Vision (ECCV)*, 2004.

[24] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *Proceedings of the 2010 European Conference on Computer Vision (ECCV'10)*, 2010.

[25] S. Shalev-Shwartz. Online learning and online convex optimization. *Found. Trends Mach. Learn.*, 2012.

[26] C. Siagian and L. Itti. Rapid biologically-inspired scene classification using features shared with visual attention. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

[27] C. Stauffer and W. E L Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, 1999.

[28] A. Torralba and A. Efros. Unbiased look at dataset bias. In *CVPR*, 2011.

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